

Lógica de Programação

CSTSI CEFET-RS

Subalgoritmos

PROBLEMA:

Escreva um programa para exibir a seguinte tela. Cada linha de asteriscos deve ser impressa com uma chamada à função **linhaDeAsteriscos()**.

```
*****
```

```
CEFET-RS
```

```
*****
```

Como permitir que a função **linhaDeAsteriscos** exiba uma quantidade **qualquer** de asteriscos?

Subalgoritmos

Tentativa 1

**Esse programa
não compila.**

Por que???

```
#include <stdio.h>
#include <stdlib.h>

void linhaDeAsteriscos(void);

main()
{
    int n;
    n=5;
    linhaDeAsteriscos();
    printf( "\nCEFET-RS\n" );
    n=22;
    linhaDeAsteriscos();
    system( "pause" );
}

void linhaDeAsteriscos()
{
    int i;

    for (i=1; i<=n; i++)
        printf( "*" );
}
```

Subalgoritmos

Tentativa 2

**Esse programa
compila, mas
não funciona!**

Por que???

```
#include <stdio.h>
#include <stdlib.h>

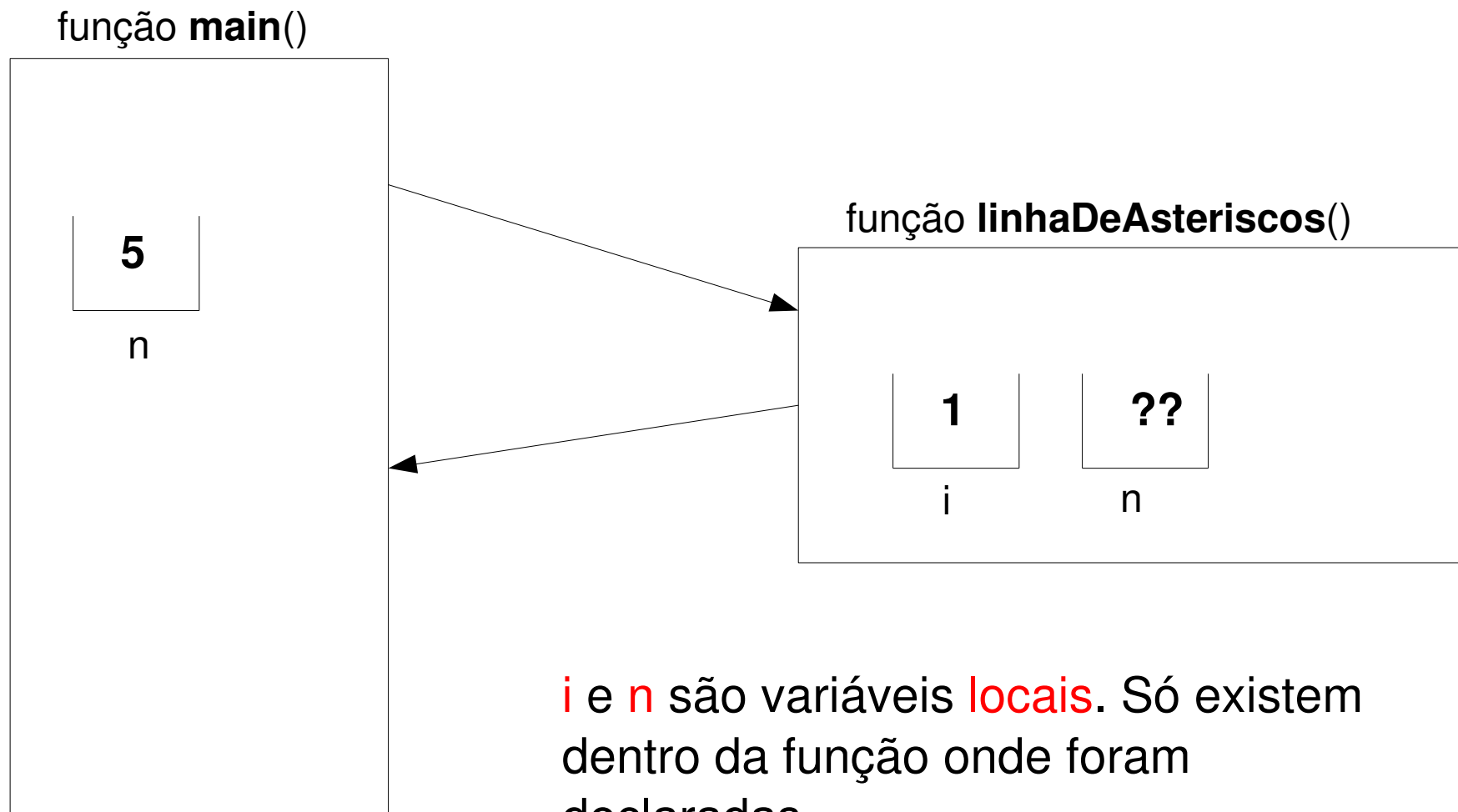
void linhaDeAsteriscos(void);

main()
{
    int n;
    n=5;
    linhaDeAsteriscos();
    printf( "\nCEFET-RS\n" );
    n=22;
    linhaDeAsteriscos();
    system( "pause" );
}

void linhaDeAsteriscos()
{
    int i,n; ←—————

    for (i=1; i<=n; i++)
        printf( "*" );
}
```

Subalgoritmos



Passagem de parâmetros

Como quebrar o
isolamento das
variáveis?

**Declarando parâmetros
de entrada.**

```
#include <stdio.h>
#include <stdlib.h>
```

```
void linhaDeAsteriscos(int n);
```

```
main()
{
    linhaDeAsteriscos(5);
    printf("\nCEFET-RS\n");
    linhaDeAsteriscos(22);
    system("pause");
}
```

```
void linhaDeAsteriscos(int n)
{
    int i;

    for (i=1; i<=n; i++)
        printf("*");
}
```

Argumento

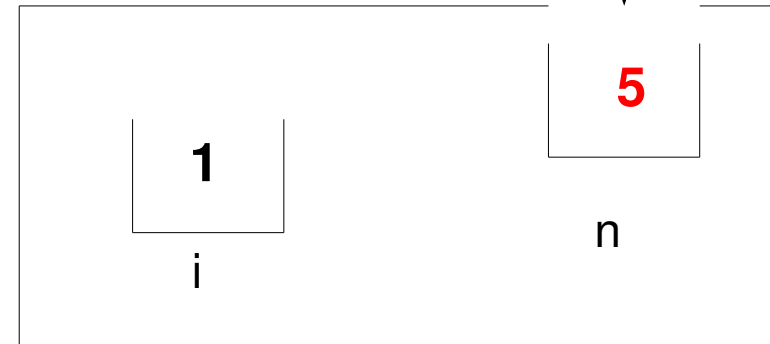


Parâmetro

Subalgoritmos

```
main()
{
  linhaDeAsteriscos(5);
  ...
}
```

linhaDeAsteriscos(int n)



O argumento **5** é passado para o parâmetro **n** declarado na função **linhaDeAsteriscos**.

A variável **n** continua existindo apenas na função onde ela foi declarada.

Subalgoritmos

PROBLEMA:

Escreva um programa para exibir a seguinte tela. Cada linha de asteriscos deve ser impressa com uma chamada à função **linhaDeAsteriscos()**.

```
*  
* *  
* * *  
* * * *  
* * * * *
```


Passagem de parâmetros

Uma variável pode ser utilizada como argumento. O **valor** da variável **a** é **copiado** para a variável **n**.

A variável **a** continua existindo apenas na função onde ela foi declarada.

```
#include <stdio.h>
#include <stdlib.h>
```

```
void linhaDeAsteriscos(int n);
```

```
main()
{
    int a;
```

```
    for (a=1; a<=5; a++) {
        linhaDeAsteriscos(a);
        printf("\n");
    }
```

```
    system("pause");
}
```

```
void linhaDeAsteriscos(int n)
{
    int i;
```

```
    for (i=1; i<=n; i++)
        printf("*");
}
```

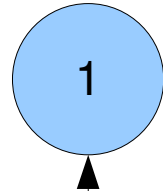
Argumento

Parâmetro

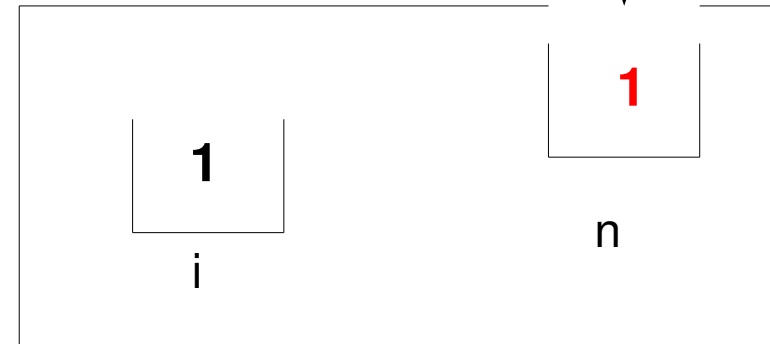
Subalgoritmos

```
main()  
{  
  int a;
```

```
  for (a=1; a<=5; a++)  
    linhaDeAsteriscos(a);  
  ...
```



linhaDeAsteriscos(int n)



Uma **cópia** da variável **a** é passada para a variável **n**.

Uma expressão também pode ser usada como argumento.

```
...  
linhaDeAsteriscos(a+4);  
...
```

Passagem de parâmetros

```
#include <stdio.h>
#include <stdlib.h>
```

O que vai ser impresso?

```
void alteraNumero(int n);
```

```
main()
{
    int num;
```

```
    num = 30;
    printf("Número (main) antes: %d\n", num);
    alteraNumero(num);
    printf("Número (main) depois: %d\n", num);
    system("pause");
}
```

```
void alteraNumero(int n)
{
    n = 40;
    printf("Número (função): %d\n", n);
}
```

Passagem de parâmetros

```
#include <stdio.h>
#include <stdlib.h>
```

O que vai ser impresso?

```
void alteraNumero(int num);
```

```
main()
{
    int num;
```

```
    num = 30;
    printf("Número (main) antes:%d\n", num);
    alteraNumero(num);
    printf("Número (main) depois:%d\n", num);
    system("pause");
}
```

```
void alteraNumero(int num)
{
    num = 40;
    printf("Número (função):%d\n", num);
}
```

Alterações



Parâmetros

Parâmetros formais

São as variáveis declaradas no cabeçalho das funções. Dentro de uma função trabalha-se com estas variáveis da mesma maneira como se trabalha com variáveis locais.

Parâmetros reais

São aqueles que substituem os parâmetros formais quando ocorre a chamada de uma função (argumentos).

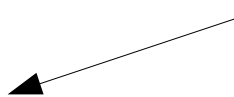
Funções com mais de um parâmetro

```
#include <stdio.h>
#include <stdlib.h>
```

```
void linhaDeCaracteres(int n,char ch);
```

```
main( )
{
    linhaDeCaracteres(8,'=');
    printf("\nCEFET-RS\n");
    linhaDeCaracteres(15,'-');
    system("pause");
}
```

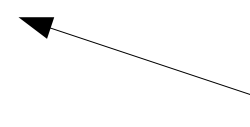
Parâmetros reais



```
void linhaDeCaracteres(int n,char ch)
{
    int i;

    for (i=1; i<=n; i++)
        printf("%c",ch);
}
```

Parâmetros formais



```
=====
CEFET-RS
-----
```

A ordem dos argumentos determina a relação entre os parâmetros reais e formais.

Parâmetros

indica que a função não possui parâmetros

```
#include <stdio.h>
#include <stdlib.h>

void linhaDezAsteriscos(void);
void exibeSoma(float a,float b);

main()
{
float x,y;

printf("Informe um valor:");
scanf("%f",&x);
printf("Informe outro:");
scanf("%f",&y);
linhaDezAsteriscos();
printf("\n");
exibeSoma(x,y);
linhaDezAsteriscos();
system("pause");
}
```

Os tipos devem ser declarados de forma independente mesmo para parâmetros de mesmo tipo

```
void linhaDezAsteriscos()
{
int i;

for (i=1; i<=10; i++)
    printf("*");
}

void exibeSoma(float a,float b)
{
float soma;

soma = a+b;
printf("Soma: %f\n",soma);
}
```