

Lógica de Programação

CSTSI CEFET-RS

PROBLEMA:

Escreva um algoritmo para exibir a seguinte tela. Cada linha que contém 22 asteriscos deve ser impressa com repetição.

```
*****
```

```
CEFET-RS
```

```
*****
```

```
CSTSI
```

```
Lógica de programação
```

```
*****
```

```
Aluno: Fulano
```

```
*****
```

Solução 1

```
i=1; i<=22; i++
```

```
Escreva "*"
```

```
Escreva "CEFET-RS"
```

```
i=1; i<=22; i++
```

```
Escreva "*"
```

```
Escreva "CSTSI"
```

```
Escreva "Lógica de programação"
```

```
i=1; i<=22; i++
```

```
Escreva "*"
```

```
Escreva "Aluno: Fulano"
```

```
i=1; i<=22; i++
```

```
Escreva "*"
```

Subalgoritmos

É um trecho de algoritmo, com uma função bem definida (o mais independente possível do restante do algoritmo) que é chamado dentro de um algoritmo principal.

Algoritmo principal

linhaDeAstericos()
Escreva "CEFET"
linhaDeAstericos()
Escreva "CSTSI"
Escreva "Lógica de programação"
linhaDeAstericos()
Escreva "Aluno: Fulano"
linhaDeAstericos()

Subalgoritmo

linhaDeAstericos()

i=1; i<=22; i++

Escreva "*"

Subalgoritmos

Importância:

- Evita que um trecho de comandos necessário em vários locais tenha que ser escrito repetidamente.
- Divide e estrutura um algoritmo em partes logicamente coerentes.
- Aumenta a legibilidade do algoritmo.
- Facilita a detecção de erros.
- Permite a reutilização de software.
- Divide um problema grande em vários menores.

Em “C” são chamados de funções

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

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

```
void linhaDeAsteriscos(void);
```

 Protótipo da função

```
main()
```

 Programa principal (função principal)

```
{
```

```
linhaDeAsteriscos();
```

 Chamada da função

```
printf("\nCEFET-RS\n");
```

```
linhaDeAsteriscos();
```

```
printf("\nCSTSI\n\n");
```

```
printf("Lógica de programação\n");
```

```
linhaDeAsteriscos();
```

```
printf("\nAluno: Fulano\n");
```

```
linhaDeAsteriscos();
```

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

```
}
```

nome da função



```
void linhaDeAsteriscos()
```

```
{
```

```
int i;
```

 Variável local

```
for (i=1; i<=22; i++)
```


```
    printf("*");
```

```
}
```

cabeçalho da função



Definição da função
(corpo)



PROBLEMA:

Escreva um programa para ler um inteiro Q e imprimir Q linhas de 22 asteriscos.

Variáveis locais:

- São visíveis apenas no local onde são declaradas.
- São criadas quando a execução da função inicia e destruídas quando termina.

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

void linhaDeAsteriscos(void);

main()
{
    int i,q;  ──────────► Variáveis locais

    printf("Informe Q:");
    scanf("%d",&q);
    for (i=1; i<=q; i++) {
        linhaDeAsteriscos();
        printf("\n");
    }
    system("pause");
}

void linhaDeAsteriscos()
{
    int i;  ──────────► Variável local

    for (i=1; i<=22; i++)
        printf("*");
}
```

Variáveis locais

Estão isoladas dentro da função onde foram declaradas.

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

```
void exhibeNumero(void);
```

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

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

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

**Esse programa
não compila.**

Por que???

Variáveis locais

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

void exhibeNumero(void);

main()
{
    int num;

    num = 30;
    exhibeNumero();
    printf("Número (main):%d\n", num);
    system("pause");
}

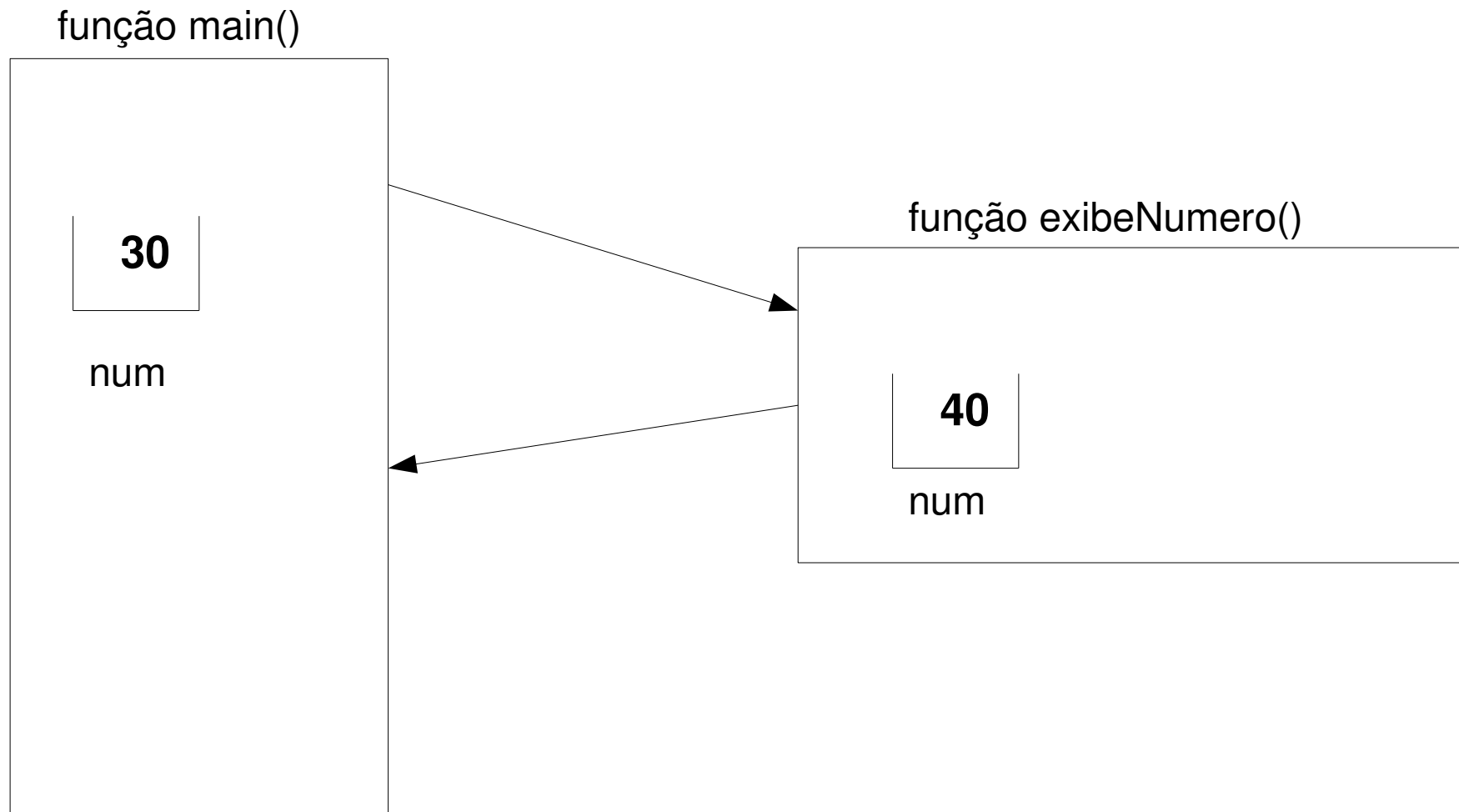
void exhibeNumero()
{
    int num;

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

“São duas variáveis (num) diferentes com o mesmo nome”

Número (função): 40
Número (main): 30

Variáveis locais



PROBLEMA:

Escreva um programa para exibir a tela abaixo. O programa deve implementar uma função chamada `retAsteriscos()` que exibe na tela um retângulo com 3 linhas e 22 colunas de asteriscos.

```
*****  
*****  
*****  
  
CEFET-RS  
  
*****  
*****  
*****
```

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

Uma função chamando outra função

```
void linhaDeAsteriscos(void);
void retAsteriscos(void);
```

```
main()
{
    retAsteriscos();
    printf("\n      CEFET\n");
    retAsteriscos();
    system("pause");
}
```

```
void linhaDeAsteriscos()
{
    int i;

    for (i=1; i<=22; i++)
        printf("*");
}
```

```
void retAsteriscos()
{
    int i;

    for (i=1; i<=3; i++) {
        linhaDeAsteriscos();
        printf("\n");
    }
}
```