

# **Programação 2**

## **Funções procedurais**

# Funções procedurais

## PROBLEMA:

Escreva um programa para exibir a seguinte tela. Cada linha que contém 10 asteriscos deve ser impressa com repetição.

```
*****  
IF-SUL  
*****  
ELETRONICA  
*****
```

# Funções procedurais

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

```
main()
```

```
{
```

```
int i;
```

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

```
printf("\n  IF-SUL\n");
```

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

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

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

```
}
```

```
*****
```

```
IF-SUL
```

```
*****
```

```
ELETRONICA
```

```
*****
```

Trecho de código repetido



Como evitar a repetição?

# Funções procedurais

Utilizando funções definidas pelo usuário.

São semelhantes a subrotinas. (CALL do assembly)

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

```
main()
```

```
{
```

```
linhaDeAsteriscos();
```

```
printf("\n  IF-SUL\n");
```

```
linhaDeAsteriscos();
```

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

```
linhaDeAsteriscos();
```

```
}
```

```
void linhaDeAsteriscos(void)
```


```
{
```

```
int i;
```

```
for (i=1; i<=10; i++)
```

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

```
}
```



O código que imprime as linhas é **definido** apenas uma vez, é **chamado** 3 vezes.

# Funções procedurais

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

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

Protótipo da função

```
main()  
{  
  linhaDeAsteriscos() ;  
  printf("\n  IF-SUL\n") ;  
  linhaDeAsteriscos() ;  
  printf("\nELETRONICA\n") ;  
  linhaDeAsteriscos() ;  
}
```

Programa principal (função principal)

```
void linhaDeAsteriscos(void)  
{  
  int i;  
  
  for (i=1; i<=10; i++)  
    printf("*") ;  
}
```

Função: **linhaDeAsteriscos**

# Funções procedurais

```
#include <stdio.h>

void linhaDeAsteriscos(void) ;

main()
{
    linhaDeAsteriscos() ;
    printf("\n  IF-SUL\n") ;
    linhaDeAsteriscos() ;
    printf("\nELETRONICA\n") ;
    linhaDeAsteriscos() ;
}

void linhaDeAsteriscos(void)
{
    int i;

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

## Importância:

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

# Funções procedurais

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

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

Protótipo da função

```
main()
```

```
{
```

```
linhaDeAsteriscos() ;
```

Chamada da função

```
printf("\n  IF-SUL\n") ;
```

```
linhaDeAsteriscos() ;
```

Chamada da função

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

```
linhaDeAsteriscos() ;
```

Chamada da função

```
}
```

nome da função

```
void linhaDeAsteriscos(void)
```

cabeçalho da função

```
{
```

```
int i;
```

Variável local

```
for (i=1; i<=10; i++)
```

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

Definição da função  
(corpo)

```
}
```

# Variáveis locais

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

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

```
main()
```

```
{
```

```
int n,j;
```

```
printf("Quantas linhas?");
```

```
scanf("%d",&n);
```

```
for (j=1; j<=n; j++)
```

```
{
```

```
linhaDeAsteriscos();
```

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

```
}
```

```
}
```

```
void linhaDeAsteriscos(void)
```

```
{
```

```
int i;
```

```
for (i=1; i<=10; i++)
```

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

```
}
```

Existe um **isolamento** das variáveis declaradas dentro de uma função.

## Variáveis locais:

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



# Isolamento das variáveis

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

Cada função possui suas próprias variáveis.

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

```
main()
```

```
{
```

```
int n,j;
```

```
printf("Quantas linhas?");
```

```
scanf("%d",&n);
```

```
for (j=1; j<=n; j++)
```

```
{
```

```
linhaDeAsteriscos();
```

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

```
}
```

```
}
```

```
void linhaDeAsteriscos(void)
```

```
{
```

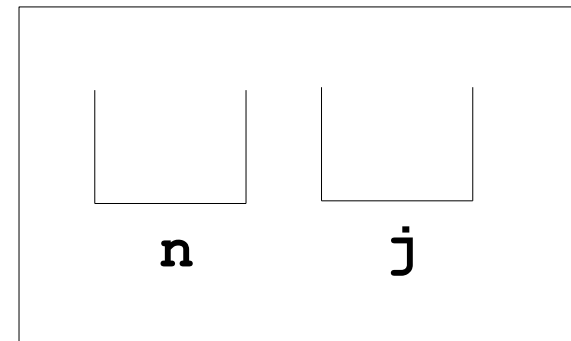
```
int i;
```

```
for (i=1; i<=10; i++)
```

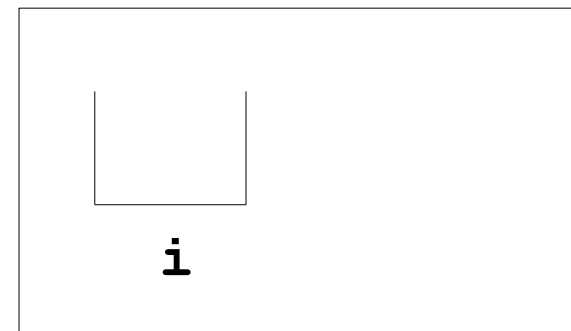
```
printf("*");
```

```
}
```

função main()



função: linhaDeAsteriscos(void)



# Variáveis locais

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

```
void incrementaX(void) ;
```

```
main()
```

```
{
```

```
int x;
```

```
x = 3;
```

```
incrementaX() ;
```

```
printf("x: %d\n",x) ;
```

```
}
```

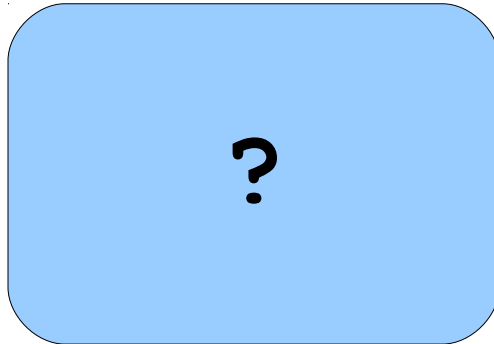
```
void incrementaX(void)
```

```
{
```

```
int x;
```

```
x++;
```

```
}
```



# Variáveis locais

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

```
void incrementaX(void);
```

```
main()
```

```
{  
  int x;
```

```
  x = 3;
```

```
  incrementaX();
```

```
  printf("x: %d\n", x);  
}
```

```
void incrementaX(void)
```

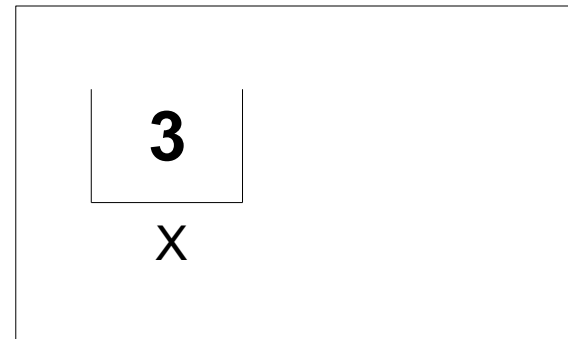
```
{  
  int x;
```

```
  x++;
```

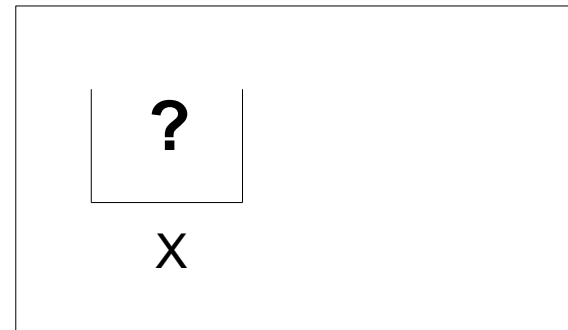
```
}
```

São duas variáveis diferentes com o mesmo nome.

função main()



função: incrementaX(void)



# Ciclo de vida das variáveis

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

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

```
main()
```

```
{
```

```
int n,j;
```

```
printf("Quantas linhas?");
```

```
scanf("%d",&n);
```

```
for (j=1; j<=n; j++)
```

```
{
```

```
linhaDeAsteriscos();
```

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

```
}
```

```
}
```

```
void linhaDeAsteriscos(void)
```

```
{
```

```
int i;
```

```
for (i=1; i<=10; i++)
```

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

```
}
```

Memória

3	n
?	j

Ponto de parada

A variável **i** ainda não existe

# Ciclo de vida das variáveis

Cada função possui suas próprias variáveis.

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

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

```
main()
```

```
{
```

```
int n,j;
```

```
printf("Quantas linhas?");
```

```
scanf("%d",&n);
```

```
for (j=1; j<=n; j++)
```

```
{
```

```
linhaDeAsteriscos();
```

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

```
}
```

```
}
```

```
void linhaDeAsteriscos(void)
```

```
{
```

```
int i;
```

```
for (i=1; i<=10; i++)
```

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

```
}
```

Memória

3	n
1	j
1	i

Ponto de parada

A variável **i** foi criada

# Ciclo de vida das variáveis

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

Cada função possui suas próprias variáveis.

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

```
main()
```

```
{
```

```
int n,j;
```

```
printf("Quantas linhas?");
```

```
scanf("%d",&n);
```

```
for (j=1; j<=n; j++)
```

```
{
```

```
linhaDeAsteriscos();
```

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

```
}
```

```
}
```

Ponto de parada

```
void linhaDeAsteriscos(void)
```

```
{
```

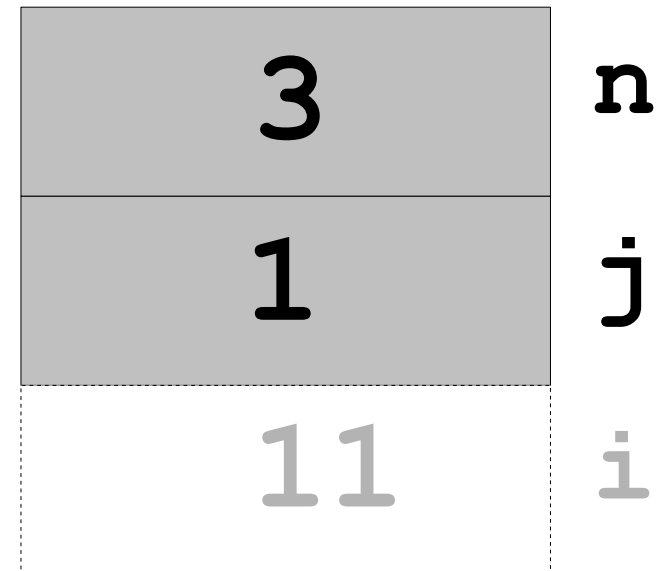
```
int i;
```

```
for (i=1; i<=10; i++)
```

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

```
}
```

Memória



A variável **i** deixa de existir

## 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 10 colunas de asteriscos.

```
*****  
*****  
*****  
ELETRONICA  
*****  
*****  
*****
```

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

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

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

Uma função chamando outra função

```
main() {
```

```
    retAsteriscos() ;
```

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

```
    retAsteriscos() ;
```

```
}
```

```
void linhaDeAsteriscos(void) {
```

```
    int i;
```

```
    for (i=1; i<=10; i++)
```

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

```
}
```

```
void retAsteriscos(void) {
```

```
    int i;
```

```
    for (i=1; i<=3; i++) {
```

```
        linhaDeAsteriscos();
```

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

```
    }
```

```
}
```