Sub-rotinas

- Modularizar um programa é dividi-lo em sub-rotinas. Estas sub-rotinas devem ser específicas, ou seja, resolver um problema apenas.
- Na linguagem C, chamamos estas sub-rotinas de funções.
- As sub-rotinas são divididas

Abaixo temos um programa com uma função main() que realiza a soma dos números 2 e 3.

```c
#include <stdio.h>

int main(){
    int a, b, resposta;

    a = 2;
    b = 3;

    resposta = a + b;

    printf("resposta = %d", resposta);

    return 0;
}
```

Vamos criar uma função que realize a soma de dois números. Sendo que, os valores a serem somados serão passados para a função através dos parâmetros.

```c
#include <stdio.h>
int soma(int x, int y){
    return x + y;
}

int main(){
    int a, b, resposta;

    a = 2;
    b = 3;

    resposta = soma(a, b);

    printf("resposta = %d", resposta);

    return 0;
}
```

Função

```
int soma(int x, int y){
    return x + y;
}
```

- ➔ x e y são parâmetros.
- ➔ Os parâmetros x e y irão receber, respectivamente os valores de a e b (variáveis do programa principal.
    - X recebe o valor de a
    - Y recebe o valor de b
    - Os tipos de dados de x e a e de y e b devem ser iguais.
- ➔ Esta função tem um tipo de retorno, ou seja, retorna um valor inteiro (resultado da soma de x + y). O **int** indica o tipo do valor de retorno.

Local onde a função está sendo chamada:

```
int main(){
    int a, b, resposta;

    a = 2;
    b = 3;

    resposta = soma(a, b);

    printf("resposta = %d", resposta);

    return 0;
}
```

A função é chamada pelo seu nome. E como a função soma possui dois parâmetros (x e y), devemos passar dois valores. Neste caso, os valores estão armazenados em a e b.

Vamos fazer uma pequena modificação do programa.

```
#include <stdio.h>
int soma(int x, int y){
    return x + y;
}

int main(){
    int resposta;

    resposta = soma(2, 3);

    printf("resposta = %d", resposta);

    return 0;
}
```

Observe que podemos passar diretamente os valores de 2 e 3 na chamada da função.

Agora analisem esta função e a sua execução:

```c
#include <stdio.h>
void soma(int x, int y){
    int r;
    r =  x + y;
}

int main(){
    int a, b, resposta;

    a = 2;
    b = 3;

    resposta = soma(a, b);

    printf("resposta = %d", resposta);

    return 0;
}
```

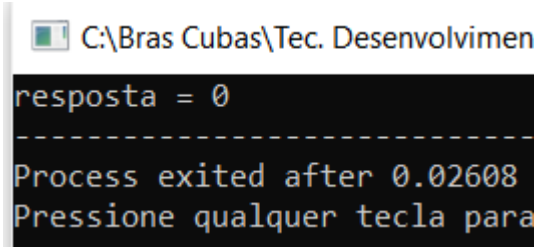
- ➔ Alteramos a função para não ter um retorno. Logo ela é void. Se definimos a função como void a palavra return não existe mais. Sendo assim, o resultado da somatória está armazenado na variável r.

```c
void soma(int x, int y){
    int r;
    r =  x + y;
}
```

- ➔ No main() temos a ocorrência de um erro na linha da chamada da função soma. Este erro está ocorrendo porque a função não pode mais retornar valor ao local de chamada. Logo, a função configurada sem retorno (void), poderá continuar sendo chamada pelo seu nome, mas não retornará o seu resultado. Veja a correção:

```c
int main(){
    int a, b, resposta;

    a = 2;
    b = 3;

    soma(a, b);

    printf("resposta = %d", resposta);

    return 0;
}
```

- ➔ O erro foi corrigido, mas o resultado da execução é este:



➔ Ainda queremos exibir o resultado da soma de 2 e 3. Vamos tentar outra coisa: Voltamos ao mesmo erro, ou seja, a função soma não retorna valor.

```c
int main(){
    int a, b;

    a = 2;
    b = 3;

    printf("resposta = %d",soma(a, b));

    return 0;
}
```

➔ Neste caso, temos que exibir o resultado da soma dentro da própria função:

```c
#include <stdio.h>
void soma(int x, int y){
    int r;
    r =  x + y;
    printf("r = %d", r);
}

int main(){
    int a, b;

    a = 2;
    b = 3;

    soma(a, b);

    return 0;
}
```

Chamada da função soma.

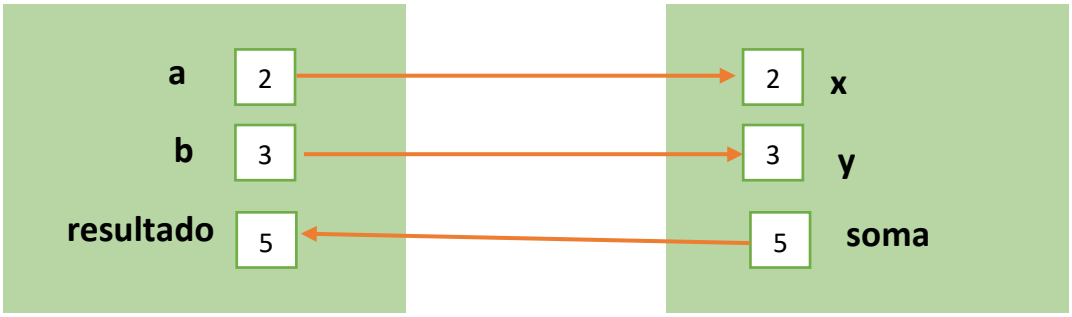
Temos dois tipos de passagem de valor por parâmetros:

- ➔ Por Valor
- ➔ Por referência

PASSAGEM DE PARÂMETRO POR VALOR

```c
#include <stdio.h>
int soma(int x, int y){
    return x + y;
}

int main(){
    int a, b, resposta;

    a = 2;
    b = 3;

    resposta = soma(a, b);

    printf("resposta = %d", resposta);

    return 0;
}
```

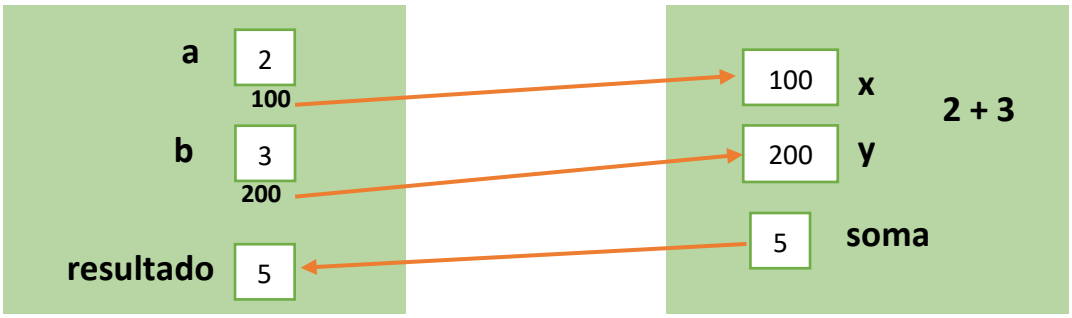
- Neste exemplo temos que será realizada uma cópia dos valores de a e b (variáveis do main) nas variáveis x e y (variáveis da função)



PASSAGEM DE PARÂMETRO POR REFERÊNCIA

```c
#include <stdio.h>
int soma(int *x, int *y){
    return *x + *y;
}

int main(){
    int a, b, resposta;

    a = 2;
    b = 3;

    resposta = soma(&a, &b);

    printf("resposta = %d", resposta);

    return 0;
}
```

- Neste exemplo as variáveis x e y recebem o endereço de memória das variáveis a e b. Logo quando realizamos a soma os valores utilizados serão aqueles que estão armazenados nas variáveis a e b.



Vamos praticar criando os programas abaixo:

1) Faça uma função que receba dois números inteiros e retorne a soma dos N números inteiros existentes entre eles.
2) Faça uma função que recebe 2 números e verifica qual o maior entre estes dois números.
3) Faça um programa que receba 3 números e usando a função do programa da questão 2 diga qual é o maior dos três números.
4) Faça um menu de opções para representar as quatro operações básicas de uma calculadora.