

# Análise e Desenvolvimento de Sistemas



## Linguagem de Programação II

### Linguagem C/C++ Função

Prof. Dr. Diego R. Moraes  
diegorafaelmoraes@gmail.com

# Função

- A **Lista\_de\_Parametros** pode ser chamada de **Lista\_de\_Argumentos**.

```
tipo_da_funcao  NomeDaFuncao  (Lista_de_Parametros)
{
    // corpo da função
}
```

# Função - Sem Parâmetro

Protótipo da função

Fundo Texto

- `system("color 3F");`
- `system("cls");`

```
#include <iostream>
#include <Windows.h> // Biblioteca para o Sleep
using namespace std;

void detalhesDoSistema();

int main() {

    detalhesDoSistema();

    system("pause");
    return 0;
}

void detalhesDoSistema() {
    int x;

    system("cls");
    system("color 3F");

    for (x = 0; x < 10; x++) {
        cout << "Contador: " << x + 1 << endl;
        cout << "Sistema: Windows 10" << endl;
        cout << "Processador: Intel I7" << endl;
        cout << "Memória: 8Gb" << endl;

        Sleep(1000);
        system("cls");
        Sleep(50);
    }
}
```

0	=	Black
1	=	Blue
2	=	Green
3	=	Aqua
4	=	Red
5	=	Purple
6	=	Yellow
7	=	White
8	=	Gray
9	=	Light Blue
A	=	Light Green
B	=	Light Aqua
C	=	Light Red
D	=	Light Purple
E	=	Light Yellow
F	=	Bright White



Hands-  
on

Hands on  
coding





# Função - Sem Parâmetro

*Exercício 3.3.* Dado um valor  $n$ , exiba uma contagem regressiva.

*Exercício 3.4.* Exiba uma tabela de conversão de polegadas em centímetros, variando as polegadas de 0 a 10 de meio em meio. [*Dica:*  $1" \approx 2,54 \text{ cm}$ ]

*Exercício 3.5.* Dados um número real  $x$  e um natural  $n$ , exiba a potência  $x^n$ .

*Exercício 3.6.* Dados um número natural  $n$ , exiba seu fatorial  $n!$ .

# Escopo de Variáveis

- Escopo de uma variável é o bloco de código onde esta variável é válida.
- Variável Local
- Variável Global

# Escopo de Variáveis - Local

X Local

```
void Funcao1();
void Funcao2();

int main(int argc, char *argv[])
{
    setlocale(LC_ALL, "Portuguese");

    int x;

    x = 100;
    printf("Valor de X dentro da função main: %d\n\n", x);

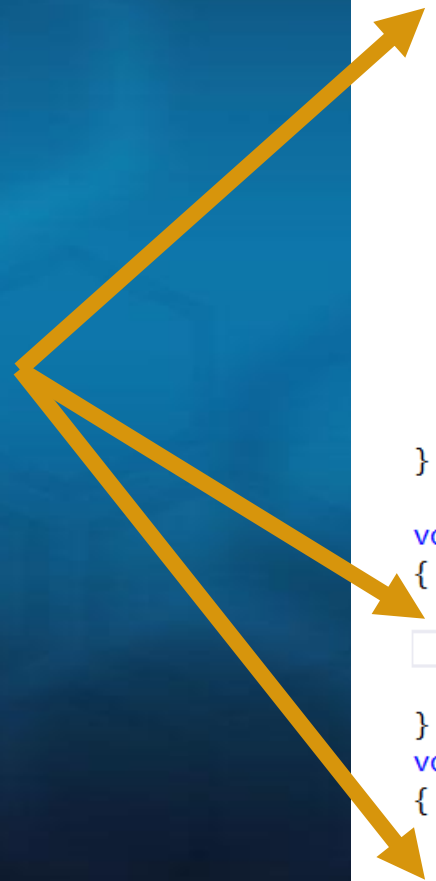
    Funcao1();
    Funcao2();

    printf("Valor de X dentro da função main: %d\n\n", x);

    system("pause");
    return 0;
}

void Funcao1()
{
    int x;
    x = 200;
    printf("Valor de X dentro da função Funcao1: %d\n\n", x);
}

void Funcao2()
{
    int x;
    x = 300;
    printf("Valor de X dentro da função Funcao2: %d\n\n", x);
}
```



# Escopo de Variáveis - Global

**X Global**

```
int x;

int main(int argc, char *argv[])
{
    setlocale(LC_ALL, "Portuguese");

    x = 100;
    printf("Valor de X dentro da função main: %d\n\n", x);

    Funcao1();
    Funcao2();

    printf("Valor de X dentro da função main: %d\n\n", x);

    system("pause");
    return 0;
}

void Funcao1()
{
    x = 200;
    printf("Valor de X dentro da função Funcao1: %d\n\n", x);
}

void Funcao2()
{
    x = 300;
    printf("Valor de X dentro da função Funcao2: %d\n\n", x);
}
```



# Escopo de Variáveis – Local e Global

**X Global**

**X Local**

```
int x;

int main(int argc, char *argv[])
{
    setlocale(LC_ALL, "Portuguese");

    x = 100;
    printf("Valor de X dentro da função main: %d\n\n", x);

    Funcao1();
    Funcao2();

    printf("Valor de X dentro da função main: %d\n\n", x);

    system("pause");
    return 0;
}

void Funcao1()
{
    x = 200;
    printf("Valor de X dentro da função Funcao1: %d\n\n", x);
}

void Funcao2()
{
    int x;
    x = 300;
    printf("Valor de X dentro da função Funcao2: %d\n\n", x);
}
```

# Passagem de Parâmetros

## Valor

- É passado uma cópia do valor da variável para a função. O valor original **NÃO PODE SER ALTERADO.**

## Referência

- É passado o endereço da variável para a função. O valor original **PODE SER ALTERADO.**

# Passagem por Valor

Funções

# Passagem de Parâmetros - Valor

```
void Funcao1(int Valor);
void Funcao2(int Valor);

int main(int argc, char *argv[])
{
    setlocale(LC_ALL, "Portuguese");
    int x;

    x = 100;

    cout << "Valor de X dentro da função main: " << x << "\n\n";

    Funcao1(x);
    Funcao2(x);

    cout << "Valor de X dentro da função main: " << x << "\n\n";

    system("pause");
    return 0;
}

void Funcao1(int valor)
{
    valor = 200;
    cout << "Valor de X dentro da função Funcao1: " << valor << "\n\n";
}

void Funcao2(int valor)
{
    valor = 300;
    cout << "Valor de X dentro da função Funcao2: " << valor << "\n\n";
}
```

# Passagem de Parâmetros - Valor

```
int main(int argc, char *argv[])  
{  
    int X;  
    X = 100;  
    Funcao1(X);  
}
```

```
void Funcao1(int Valor)  
{  
    Valor = 200;  
}
```



# Passagem de Parâmetros - Valor

```
int main(int argc, char *argv[])
```

```
{
```

```
    int X;
```

```
    X = 100;
```

```
    Funcao1( Envia uma CÓPIA
```

```
}
```

```
void Funcao1(int )
```

```
{
```

```
    Valor = 200;
```

```
}
```

# Passagem de Parâmetros - Valor

```
int main(int argc, char *argv[])  
{  
    int X;  
    X = 100;  
    Funcao1(100)  
}
```

```
void Funcao1(int 100)  
{  
    Valor = 200;  
}
```

# Passagem de Parâmetros - Valor

```
int main(int argc, char *argv[])  
{  
    int X;  
    X = 100;  
    Funcao1(100)  
}
```

```
void Funcao1(int Valor)  
{  
    200 Valor = 200;  
}
```



Hands-  
on

Hands on  
coding



# Exercício

- Solicite 4 notas de um aluno na tela. Crie uma função que receba estas notas e retorne a média entre elas.
- Dados dois números distintos, crie uma função que receba estes números e retorne o maior. Na sequência, exiba o número retornado na tela.



# Exercício

- Faça funções para:

*Exercício 2.2.* Dados dois números distintos, informe qual dele é o maior.

*Exercício 2.3.* Dado um ano, informe se ele é ou não bissexto. [*Dica:* um ano é bissexto se é divisível por 4 mas não por 100].

*Exercício 1.6.* Dada uma temperatura em graus *Fahrenheit*, informe o valor correspondente em graus *Celsius*. [*Dica:*  $C = (F - 32) * (5 / 9)$ ].

*Exercício 1.4.* Dados uma distância e o total de litros de combustível gasto por um automóvel para percorrê-la, informe o consumo médio.

# Exercício

```
void anoBissexto(int ano);

int main() {
    int ano;

    cout << "Informe um ano: ";
    cin >> ano;

    anoBissexto(ano);

    system("pause");
    return 0;
}

void anoBissexto(int ano) {
    int divisivelPor4, divisivelPor100;

    divisivelPor4 = ano % 4;
    divisivelPor100 = ano % 100;

    // Bissexto: divisível por 4 mas NÃO é divisível por 100
    if ((divisivelPor4 == 0) && (divisivelPor100 != 0)) {
        cout << "O ano é bissexto" << endl << endl;
    }
    else {
        cout << "O ano NÃO é bissexto" << endl << endl;
    }
}
```

# Passagem por Referência

Funções

# Passagem de Parâmetros – Referência SEM PONTEIROS

Utilizando  
&

```
//REFER.CPP
//Mostra passagem de argumentos por referência
#include <iostream.h>

void reajusta20( float& p, float& r);

void main()
{
    float preco,val_reaj;

    do
    {
        cout << "\n\nInsira o preço atual: ";
        cin  >> preco;
        reajusta20(preco,val_reaj);
        cout << "Preço novo          = " << preco
              << "\nAumento          = " << val_reaj;
    } while( preco != 0.0);
}

//reajusta20()
//Reajusta o preço em 20%
void reajusta20(float& p, float& r)
{
    r = p * 0.2;
    p *= 1.2;
}
```

# Passagem de Parâmetros – Referência COM PONTEIROS

Utilizando



```
//PONTEIRO.CPP
//Mostra passagem de argumentos por referência com ponteiros
#include <iostream.h>

void reajusta20( float *p, float *r);

void main()
{
    float preco,val_reaj;

    do
    {
        cout << "\n\nInsira o preço atual: ";
        cin  >> preco;
        reajusta20(&preco,&val_reaj);
        cout << "Preço novo          = " << preco
              << "\nAumento          = " << val_reaj;
    } while( preco != 0.0);
}

//reajusta20()
//Reajusta o preço em 20%
void reajusta20(float *p, float *r)
{
    *r = *p * 0.2;
    *p *= 1.2;
}
```



# Passagem de Parâmetros - Referência

```
int main(int argc, char *argv[])  
{  
    int X;  
    X = 100;  
    Funcao2(&X);  
}
```

```
void Funcao2(int *Valor)  
{  
    *Valor = 300;  
}
```

# Passagem de Parâmetros - Referência

```
int main(int argc, char *argv[])
```

```
{
```

```
    int X;
```

```
    X = 100;
```

```
    Funcao2( Envia o ORIGINAL
```

```
}
```

```
void Funcao2(int *r)
```


```
{
```

```
    *Valor = 300;
```

```
}
```

# Passagem de Parâmetros - Referência

```
int main(int argc, char *argv[])  
{  
    int X;  
    X = 100;  
    Funcao2(&X);  
}
```

```
void Funcao2(int *r)  
{  
    *Valor = 300;  
}
```

# Passagem de Parâmetros - Referência

```
int main(int argc, char *argv[])  
{  
    int X;  
    X = 100;  
    Funcao2(300)  
}
```

```
void Funcao2(int *Valor)  
{  
    300lor = 300;  
}
```

# Passagem de Parâmetros - Referência

```
int main(int argc, char *argv[])  
{  
    int X;  
    X = 100;  
    Funcao2(&X);  
}
```

```
void Funcao2(int *Valor)  
{  
    *Valor = 300;  
}
```



# Passagem de Parâmetros - Referência

```
void Funcao1(int Valor);
void Funcao2(int *valor);

int main(int argc, char *argv[])
{
    setlocale(LC_ALL, "Portuguese");
    int x;

    x = 100;
    cout << "\tMain ==> Valor de X ANTES das funções: " << x << "\n\n";

    Funcao1(x);
    cout << "\tMain ==> Valor de X depois da Função1: " << x << "\n\n";

    Funcao2(&x);
    cout << "\tMain ==> Valor de X depois da Função2: " << x << "\n\n";

    system("pause");
    return 0;
}

void Funcao1(int valor)
{
    valor = 200;
    cout << "\tValor de X dentro da função Funcao1: " << valor << "\n";
}

void Funcao2(int *valor)
{
    *valor = 300;
    cout << "\tValor de X dentro da função Funcao2: " << *valor << "\n";
}
```

# Exemplo

```
void troca_valores(int *ptrNro1, int *ptrNro2);

int main() {

    int nro1, nro2;
    cout << "Digite o primeiro valor: ";
    cin >> nro1;

    cout << "Digite o segundo valor: ";
    cin >> nro2;

    cout << "Voce digitou os valores na seguinte ordem: " << nro1 << " e " << nro2 << endl;

    troca_valores(&nro1, &nro2);

    cout << "Os valores trocados sao: " << nro1 << " e " << nro2 << endl;

    system("pause");
    return 0;
}

void troca_valores(int *ptrNro1, int *ptrNro2)
{
    int auxiliar;

    auxiliar = *ptrNro1;

    *ptrNro1 = *ptrNro2;

    *ptrNro2 = auxiliar;
}
```

# Exemplo

```
void separarValores(float nro, int *parteInt, float *parteFrac);

int main() {
    float nro1, parteFracionada;
    int parteInteira;

    cout << "Digite um nro com casas decimais: ";
    cin >> nro1;

    separarValores(nro1, &parteInteira, &parteFracionada);

    cout << "Parte inteira: " << parteInteira << endl;
    cout << "Parte fracionada: " << parteFracionada << endl;

    system("pause");
    return 0;
}

void separarValores(float nro, int *parteInt, float *parteFrac) {
    *parteInt = (int)nro;
    *parteFrac = nro - *parteInt;
}
```



Hands-  
on

Hands on  
coding



# Exercício

- Fazer uma função para ler e retornar o valor das 4 notas de um aluno e sua média.
- Fazer uma função para ler e retornar o número de uma sala de aula, sua capacidade e o total de alunos matriculados
- Fazer uma função que recebe um número, calcule e retorne se o número é positivo ou negativo e se é múltiplo ou não de 3.
- Fazer uma função que retorna a soma, a diferença e o produto entre dois números.

Ex: `void calculo(int nro1, int nro2, int *soma, int *dif, int *prod)`



# Análise e Desenvolvimento de Sistemas



## Linguagem de Programação II

### Linguagem C/C++ Função

Prof. Dr. Diego R. Moraes  
diegorafaelmoraes@gmail.com