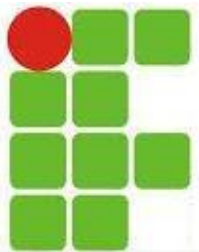


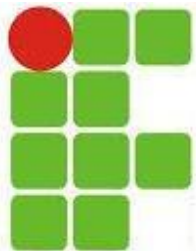
# ALGORITMOS

**Professor: Diego Oliveira**



**Conteúdo 09:**  
**Funções**

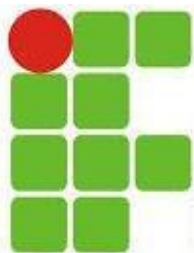




# Funções

- Até agora, todo o código desenvolvido ficou dentro da função `main()`
- Porém pode-se criar quantas funções forem necessárias para resolver um problema
- O exemplo mais clássico é a calculadora com as 4 operações básicas:
  - `soma()`
  - `subtracao()`
  - `multiplicacao()`
  - `divisao`

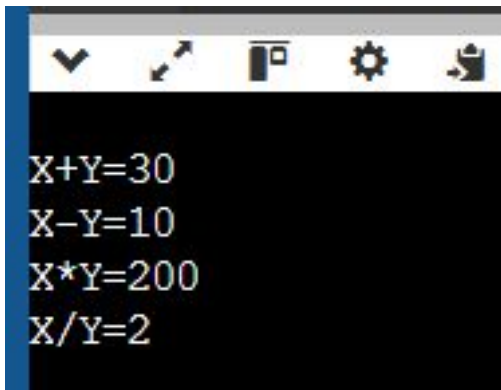




# Exemplos de Funções

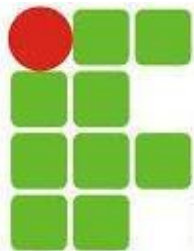
```
1  #include <stdio.h>
2
3  int x = 20;
4  int y = 10;
5
6  void soma() {
7      printf("X+Y=%d\n", x + y);
8  }
9
10 void subtracao() {
11     printf("X-Y=%d\n", x - y);
12 }
13
14 void multiplicacao() {
15     printf("X*Y=%d\n", x * y);
16 }
17
18 void divisao() {
19     printf("X/Y=%d\n", x / y);
20 }
21
22 int main() {
23     soma();
24     subtracao();
25     multiplicacao();
26     divisao();
27     return 0;
28 }
```

SAÍDA



```
X+Y=30
X-Y=10
X*Y=200
X/Y=2
```

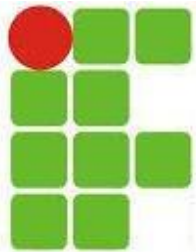




# Funções

- No slide anterior temos:
  - O modificador de acesso **public** permite que a função seja utilizada por outras classes (assunto de POO)
  - O tipo de retorno **void** significa que esta função não retorna nada. Pode ser trocado por **int**, **double**, **String**, **boolean**...
  - O nome **soma** representa o nome da função
  - Os **parênteses vazios** significam que esta função não recebe parâmetros, ou seja, valores de quaisquer tipos
  - As VARIÁVEIS que podem ser criadas fora das funções, assim elas terão um escopo GLOBAL, ou seja, podem ser acessadas de todas as funções
- Observe que as funções são chamados dentro do MAIN para serem executadas!

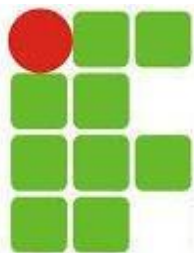




# Funções

- Observe que as 4 funções criados foram chamados dentra da função **main()**
- As 4 funções são **void**, não retornam nada
- Pode-se estabelecer um tipo de retorno, como **int**, e fazer a impressão no próprio MAIN vejamos:





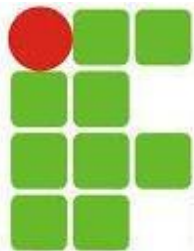
# Retorno de Funções

```
1  #include <stdio.h>
2
3  int x = 50;
4  int y = 10;
5
6  int soma() {
7      return x + y;
8  }
9
10 int subtracao() {
11     return x - y;
12 }
13
14 int multiplicacao() {
15     return x * y;
16 }
17
18 int divisao() {
19     return x / y;
20 }
21
22 int main() {
23     printf("X+Y=%d\n", soma());
24     printf("X-Y=%d\n", subtracao());
25     printf("X*Y=%d\n", multiplicacao());
26     printf("X/Y=%d\n", divisao());
27     return 0;
28 }
29
```

SAÍDA

```
X+Y=60
X-Y=40
X*Y=500
X/Y=5
```

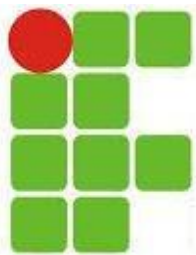




# Funções

- No segundo exemplo mostrado, vimos que é possível chamar as funções e obter um retorno destas funções
- Porém este retorno é sempre o mesmo, pois os valores são fixos (X e Y)
- E se quisermos efetuar os cálculos com os nossos próprios valores?
- Para isso utilizaremos os parâmetros!





# Parâmetros de Funções

```
1  #include <stdio.h>
2
3  int soma(int x, int y) {
4      return x + y;
5  }
6
7  int subtracao(int x, int y) {
8      return x - y;
9  }
10
11 int multiplicacao(int x, int y) {
12     return x * y;
13 }
14
15 int divisao(int x, int y) {
16     return x / y;
17 }
```

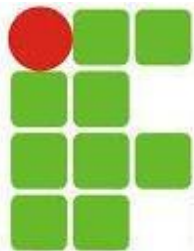
```
19 int main() {
20     int x, y;
21
22     printf("Digite o valor de x: ");
23     scanf("%d", &x);
24
25     printf("Digite o valor de y: ");
26     scanf("%d", &y);
27
28     printf("X+Y=%d\n", soma(x, y));
29     printf("X-Y=%d\n", subtracao(x, y));
30     printf("X*Y=%d\n", multiplicacao(x, y));
31     printf("X/Y=%d\n", divisao(x, y));
32
33     return 0;
34 }
```



**SAÍDA**

```
✓ ↗ 📄 ⚙️ 🗑️
X*Y=800
X/Y=2
```

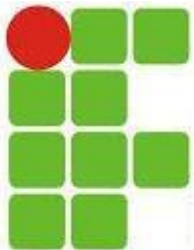




# Exercício

- Escreva uma função que imprima uma matriz de zeros de X por Y, recebidos por parâmetro. Esta função **não retorna nada**.
- Escreva uma função que receba um número e **retorne** uma String dizendo se o número é PAR ou ÍMPAR
- Escreva uma função que receba 3 números inteiros X, Y e Z e **retorne** qual o maior deles





# Perguntas?

