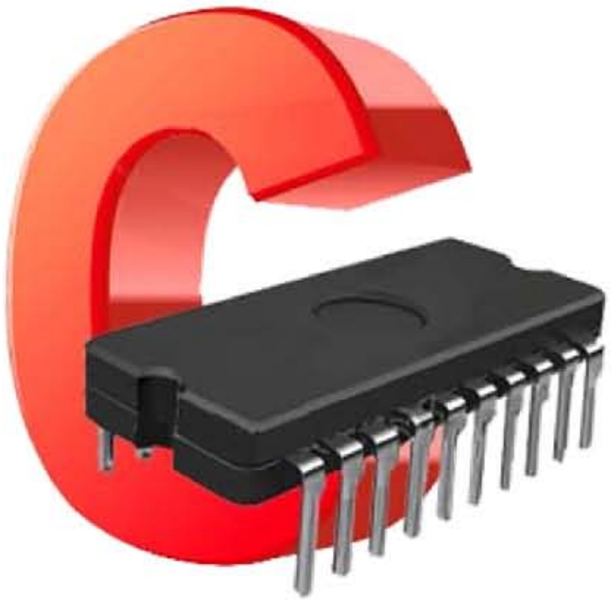
A LINGUAGEM C



para PIC

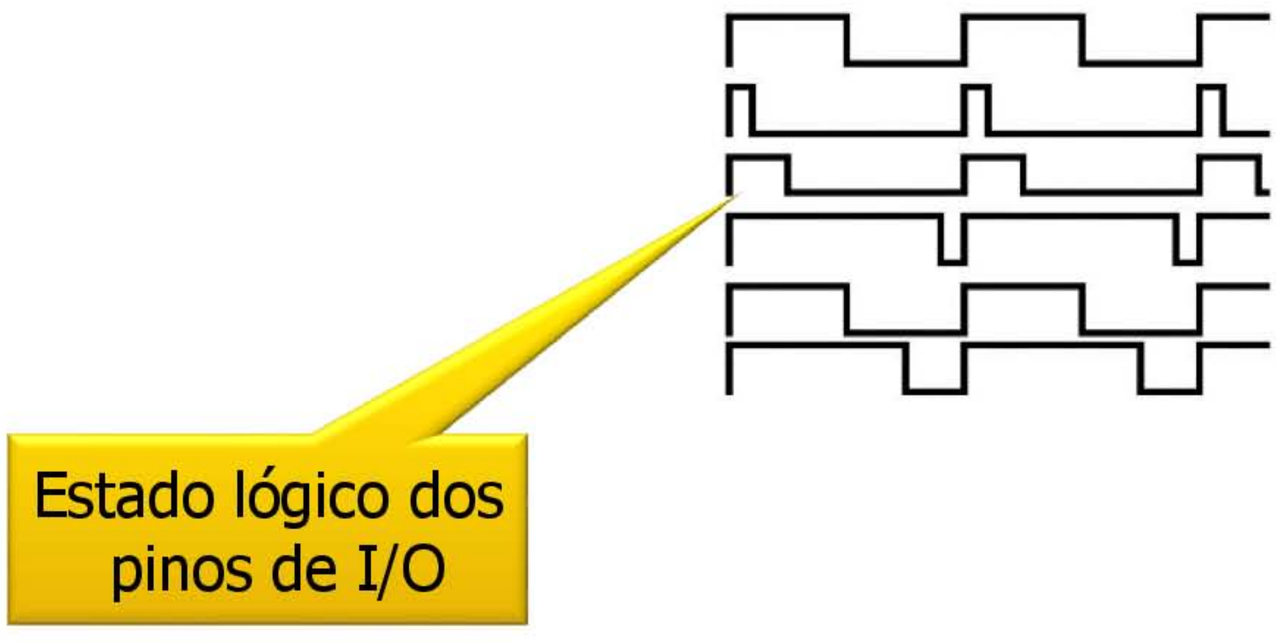
© Prof. Eng° esp Luiz Antonio Vargas Pinto
www.vargasp.com

Leitura de dados

- Entrada de dados é feita pela leitura do estado lógico de um pino específico de uma porta.
- Formato Geral:

```
variável_destino = PORTC.RC7;
if(variável_destino == 0)
```

  - (Sendo PORT de A...E e pinos de 0...7)



© Prof. Vargas

Escrita de dado

- Saída de dados é feita pela escrita do estado lógico em um pino específico de uma porta.
- Formato Geral:

```
PORTA.RA1 = 1; // ou 0
LATA.RA1 = 1; // ou 0
```

PORTA

Pino de I/O

Estado lógico do pino de I/O

© Prof. Vargas

Comando if

Forma:

```
if (condição)
{
    Comando 1;
}
```

ou:

```
if (condição)
{
    Comando 1;
}else
{
    Comando 2;
}
```

© Prof. Vargas

Comando if

```
#define Sen_Emerg RA5_bit

if(Sen_Emerg==0)
{
    goto emergencia;
}
```

Declaração do bit

Label

© Prof. Vargas

Condicionais

- **==** igual
- **!=** diferente de
- **>** maior que
- **<** menor que
- **>=** maior ou igual
- **<=** menor ou igual.

© Prof. Vargas

Um programa

```
/*-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
 Programa: nome.c
 Data    : DD/MM/AAAA
 Autor   : nome
 Descrição:
 -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-*/
void main(void)
{
    PORTB = 0xFF; // PORTB = 1111.1111 todos LEDs off
    TRISB = 0x80; // TRISB = 1000.0000
    for(;;)
    {


    }
}
```

Descrição do programa

Comentário

Economiza espaço

Loop permanente

© Prof. Vargas

Operadores lógicos e aritméticos

© Prof. Vargas

Características

Operador	Ação
+	Soma inteira e fracionada
-	Subtração e troca de sinais
*	Multiplicação inteira ou fracionada
/	Divisão inteira e fracionada
%	Resto da divisão
++	Incremento
--	decremento

Operador	Ação
&&	and
||	or
!	not

© Prof. Vargas

Aritméticos

- Operadores de incremento e decremento são unitários e alteram a variável sobre a qual estão aplicados.

```
x++;                    x--;
```

são equivalentes a:

```
x=x+1;                  x=x-1;
```

- Podem ser pré-fixados ou pós- fixados.
  - A diferença é: **pré-fixados** incrementam e retornam o valor da variável já incrementada. **Pós-fixados** retornam o valor da variável sem o incremento e depois de usados incrementam a variável.

© Prof. Vargas

Laços de repetição

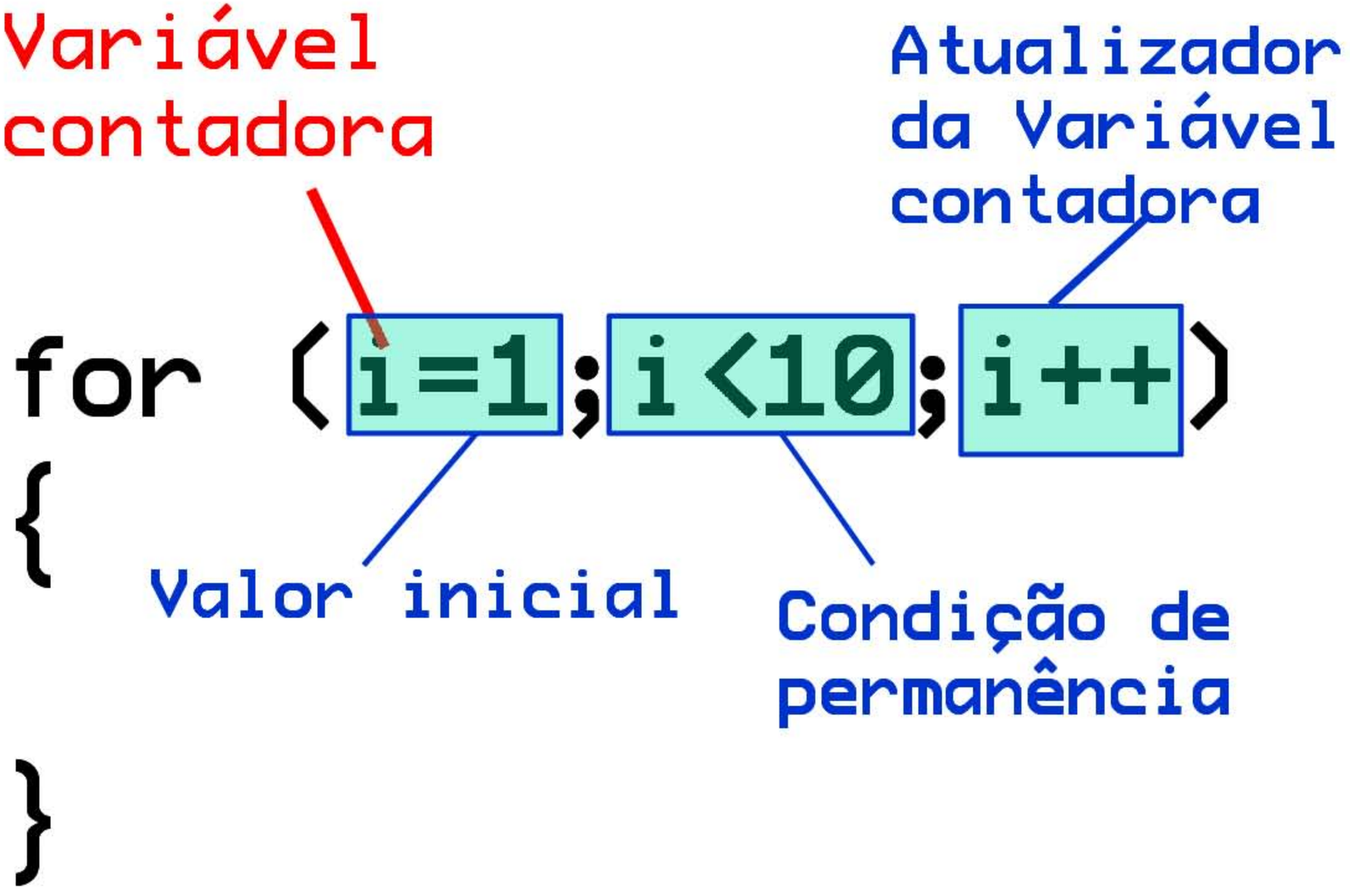
© Prof. Vargas

Componentes

- Valor inicial
  - Onde começar
- Contador de repetição
  - Controla a quantidade de laços
  - Variável contadora
- Condições de encerramento
  - Teste lógico

© Prof. Vargas

Função **for**



© Prof. Vargas

Função **while**

Valor inicial

Condição de permanência

```
i=0;
while(i<10)
{
    i++;
}
```

Variável contadora

Atualizador da Variável contadora

ou i=i+1;

© Prof. Vargas

Função **do...while**

Valor inicial

```
i=0;
do
{
    i++;
}while(i<10);
```

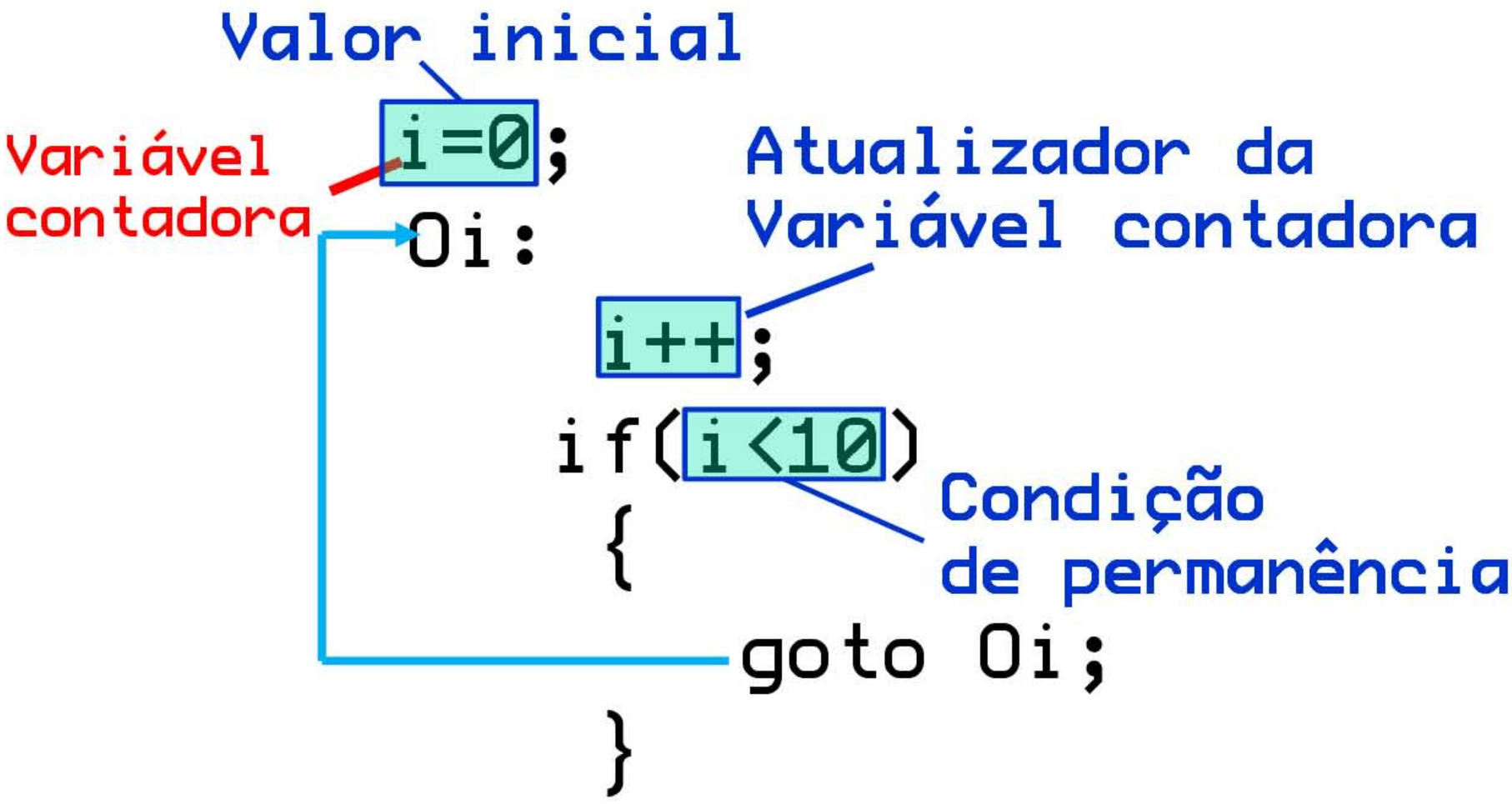
Atualizador da Variável contadora

Variável contadora

Condição de permanência

© Prof. Vargas

Função **goto**



© Prof. Vargas

Funções

© Prof. Vargas

Funções

- Função é um bloco de código de programa que pode ser usado diversas vezes em sua execução.
- O uso de funções permite que o programa fique mais legível, mais bem estruturado.
- No fundo, um programa C consiste de várias funções colocadas juntas.
- Sintaxe da declaração:

```
tipo nome_da_função (lista_de_parâmetros)
{
    comando1;
    comando2;
}
```

© Prof. Vargas

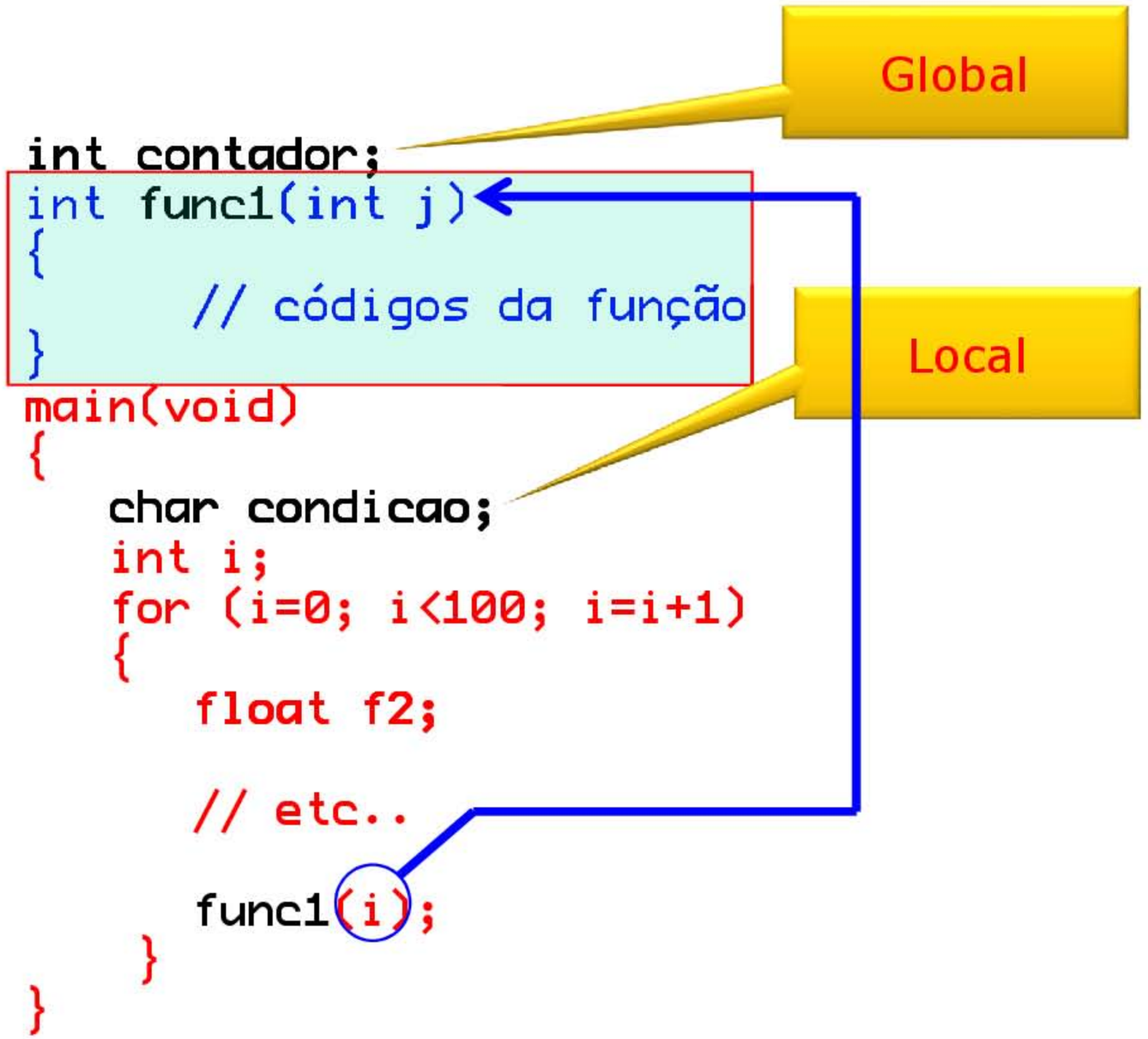
Sintaxe

```
void nome_da_função (parâmetros);

tipo nome_da_função (void);

void nome_da_função (void);
```

- O **void** é o tipo vazio. A aplicação deste tipo será vista posteriormente.

© Prof. Vargas

Exemplo



© Prof. Vargas

Exemplo clássico

```
int mult (float a, float b, float c)  // Multiplica 3
números
{
  printf ("%f",a*b*c);
  return(0);
}
main (void)
{
   float x,y;
   x=23.5;
   y=12.9;
   mult (x,y,3.87);
   system("pause");
}
```

Tem comentários

© Prof. Vargas

Análise

- x → a
- y → b
- 3.87 → c
- Não existe retorno para as variáveis originantes pois temos `return(0)` o que as mantém com seus valores originais

© Prof. Vargas

Com retorno de valores

x recebe 12
Y recebe 7

```
int prod (int x,int y)
{
   return (x*y);
}

main (void)
{
   int saida;
   saida = prod (12,7);
   if (saida>100 LATB.RB2 = 0;
}
```

Local

Veja a forma

© Prof. Vargas

Análise

1. Se não especificamos o tipo de retorno o compilador considera este tipo inteiro – o que não é boa prática.
2. **main** sempre retorna **int**.
3. Idealmente usaremos `return(0);`
4. O tipo de cada receptor deve ser especificado na função.

© Prof. Vargas

Protótipo de função

- Protótipos são opcionais, mas são exigidos pela sucessora C++.
- Protótipos permitem que C forneça uma verificação mais forte de tipos.

© Prof. Vargas

exemplo

```
int prod (int x,int y);

main (void)
{
   int saida;
   saida = prod (12,7);
   if (saida>100) LATB.RB1 = 1;

}
```

Prototipagem

```
int prod (int x,int y)
{
   return (x*y);
}
```

Código real

© Prof. Vargas