

# Linguagem de Programação

(ILP-010)

---

Prof. Dr. Silvio do Lago Pereira

Departamento de Tecnologia da Informação

Faculdade de Tecnologia de São Paulo



## Contato

- **Sala:** 623 – Bloco A
- **E-mail:** slago@ime.usp.br
- **Página:** [www.ime.usp.br/~slago](http://www.ime.usp.br/~slago)
  - Ementa
  - Compilador Pelles C
  - Bibliografia
  - Critérios de avaliação
  - Notas
  - Cronograma de aulas e provas



# Curso

- **Objetivo:** Implementar algoritmos em linguagem C.
- **Tópicos:**
  - E/S básica
  - Estruturas de controle
  - Funções
  - Vetores
  - Estruturas
  - Ponteiros
  - Arquivos



# Avaliação

- **Provas e trabalhos**

- T1
- P1
- T2
- P2
- SUB

- **Média** =  $0.4 * (P1 + P2) + 0.1 * (T1 + T2)$

- Aprovação requer média maior ou igual a 6,0.

- **Prova substitutiva**

- Apenas para quem não atingir a média
  - Substitui a menor nota de prova (apenas uma delas)

# Introdução

## (ILP-010)





# Introdução

## Linguagem C

é considerada uma das linguagens de programação mais eficientes que existem.

### Histórico:

- Criada em 1972, por Dennis Ritchie, nos Laboratórios Bell.
- Evolução da linguagem BCPL ( $\rightarrow$  B  $\rightarrow$  C  $\rightarrow$  C++, JAVA, C#, ...).

### Algumas características:

- **Flexível**: usada na criação de diversos tipos de aplicações.
- **Portátil**: pode ser executada em diferentes plataformas.
- **Eficiente**: proporciona velocidade de execução e economia de memória.

### Algumas aplicações:

- Jogos.
- Controladores.
- Compiladores.
- Sistemas operacionais.





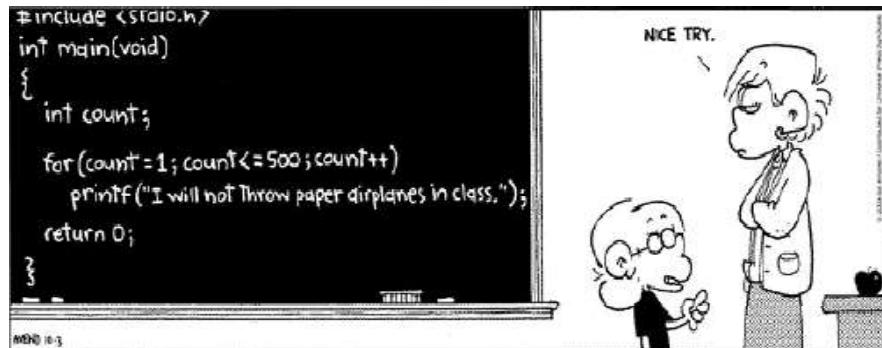
# Introdução

## Um programa em C

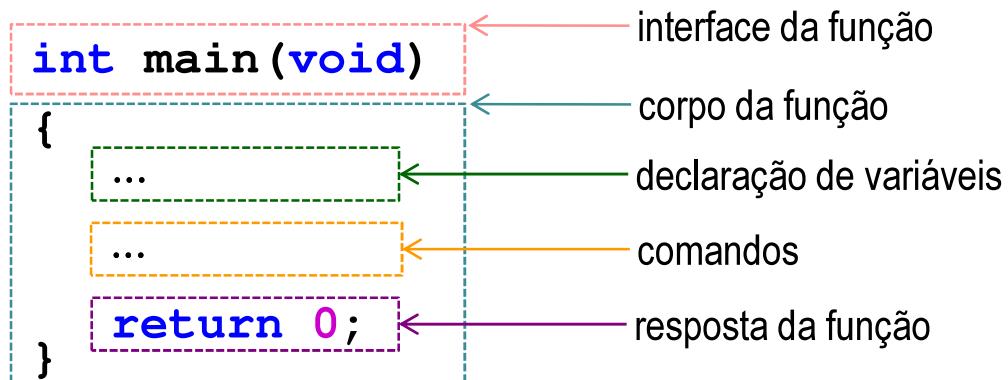
é um conjunto de uma ou mais funções, em que uma delas é chamada **main**.

### Função:

- Bloco de código independente e reusável.
- A função principal em C é chamada **main**.
- A execução sempre inicia na função principal.



### Função principal:





# Introdução

## Exemplo 1. Índice de massa corpórea (IMC)

Uma pessoa está obesa se seu IMC (peso dividido pela altura ao quadrado) é superior a 30. Dados o peso e a altura de uma pessoa, informe se ela está obesa.

```
/* OBESO.C - determina se uma pessoa está obesa */
#include <stdio.h> // entrada e saída padrão
#include <math.h> // funções matemáticas

int main(void) {
    float peso, altura, imc;
    printf("Qual o peso e a altura? ");
    scanf("%f %f", &peso, &altura);
    imc = peso/pow(altura,2);
    printf("IMC = %.1f\n", imc);
    if( imc<=30 ) printf("Nao esta obesa!\n");
    else printf("Esta obesa!\n");
    return 0;
}
```



# Introdução

## Algumas observações importantes:

- C permite comentários de várias linhas (`/* e */`) e de uma única linha (`//`).
- A diretiva `#include` inclui arquivos com declarações necessárias à compilação.
- O arquivo `stdio.h` (*standard input/output header*) declara funções de E/S padrão.
- O arquivo `math.h` (*mathematical functions header*) declara funções matemáticas.
- Letras maiúsculas e minúsculas são consideradas distintas.
- Variáveis devem ser *declaradas* e *iniciadas* antes de serem usadas.
- O tipo `float` é usado para declarar variáveis que guardam números reais.
- A função `printf()` exibe dados no vídeo e a função `scanf()` lê dados do teclado.
- Cálculos são feitos com operadores aritméticos (e.g., `/`) e funções matemáticas (e.g., `pow`).
- A atribuição de valores às variáveis é feita pelo operador de atribuição `=`.
- O `if-else` executa um entre dois comandos alternativos, dependendo de uma comparação.
- Comparações são feitas com operadores relacionais convencionais (e.g., `<=`).
- O comando `return` termina a execução da função e devolve seu resultado.



# Introdução

## Compilador *Pelles C*

é o software que usaremos para criar e executar programas codificados em C!

The screenshot shows the Pelles C IDE interface. The main window displays the source code for 'obesa.c'. The code is a simple C program that calculates the IMC (Body Mass Index) and checks if a person is obese based on the IMC value. The IDE has a toolbar, a menu bar with File, Edit, View, Project, Source, Tools, Window, Help, and a status bar at the bottom.

```
/* OBESO.C - determina se uma pessoa está obesa */

#include <stdio.h> // entrada e saída padrão
#include <math.h> // funções matemáticas

int main(void) {
    float peso, altura, imc;
    printf("Qual o peso e a altura? ");
    scanf("%f %f", &peso, &altura);
    imc = peso/pow(altura,2);
    printf("IMC = %.1f\n",imc);
    if( imc<=30 ) printf("Nao esta obesa!\n");
    else printf("Esta obesa!\n");
    return 0;
}
```

### Criação de programa:

- Inicie a execução do **Pelles C IDE**.
- No menu, selecione **File** → **New** → **Project** → **Win32 Console program**.
- Dê um nome ao projeto e clique **OK**.
- Pressione **Ctrl-N** para criar o arquivo.
- Digite o código do programa.
- Pressione **Ctrl-S** para salvar.
- Pressione **Ctrl-F5** para executar.

Esse compilador é de uso livre e está disponível para *download* na página da disciplina!



# Introdução

## Exercício 1. Índice de massa corpórea

Execute o programa **obeso.c**, usando o compilador *Pelles C*.

## Exercício 2. Perímetro da circunferência

Corrija os erros existentes no programa a seguir:

```
/* PERIM.C - informa o perímetro da circunferência */
#include <studio.h>

Main() {
    float raio;
    printf("Qual o raio? "); /* solicita o raio
    scanf("%f",raio);           da circunferência */
    float perim = 2*3.14*raio; /* calcula o perímetro */
    printf("Perímetro = %f\n",perim)
    return;
}
```



# Introdução

## Tipo de dados

define um conjunto de **valores** e um conjunto de **operações** que podem ser feitas com eles.

### Tipos básicos em C:

Tipo	Bytes	Escala
<b>char</b>	1	<b>-128 ... +127</b>
<b>int</b>	4	<b>-2147483648 ... +2147483647</b>
<b>float</b>	4	<b>3.4e-38 ... 3.4e+38</b> (absoluto)
<b>double</b>	8	<b>1.7e-308 ... 1.7e+308</b> (absoluto)
<b>void</b>	0	vazia

### Um dado pode ser:

- **Constante**: cujo tipo depende de como ela é representada.
- **Variável**: cujo tipo depende de como ela é declarada.



# Introdução

## Tipos e constantes:

- **char** (caractere) :
  - 'a', '2', '@', '\n', ...
- **int** (número inteiro) :
  - Decimal: 1, 234, 56, ...
  - Octal: 05, 067, 012, ...
  - Hexadecimal: 0x1, 0x7F, 0x8A, ...
- **double** (número real) :
  - Notação comum: 0.1, 2.34, 5.0, ...
  - Notação científica: 5.67e2, 1e3, 0.1e4, ...
- Não há o tipo **string** (cadeia de caracteres) em C, mas há constantes desse tipo:
  - "b", "1", "Ana", ...

Em C, não há distinção entre caracteres e seus respectivos códigos ASCII!



# Introdução

## Tipos e variáveis:

- Em C, toda variável deve ser **declarada** e **iniciada** antes de ser usada.
- A **declaração** de uma variável consiste de um tipo seguido de um identificador.
- Um **identificador** é um nome iniciando com letra (maiúscula ou minúscula), ou sublinha, e deve ser composto exclusivamente por letras, dígitos e sublinhas.
- O tipo determina a **quantidade de memória** que deve ser alocada para a variável.
- O identificador permite **citar** a variável no bloco no qual ela é declarada.
- Uma variável pode ser **iniciada** ao ser declarada (com o operador de atribuição **=**).

## Exemplo 2. Declaração de variável

```
char tecla='A', opcao;  
int x, y=3, z;  
float comissao=0.10, desconto, salario;
```

Em C, não é possível declarar uma variável simples do tipo **void**!



# Introdução

## Modificadores de tipo

- **signed**: indica que uma variável guardará inteiros **com** sinal (**char** ou **int**).
- **unsigned**: indica que uma variável guardará inteiros **sem** sinal (**char** ou **int**).
- **short**: pode reduzir à metade o espaço de memória usado por um número (**int**).
- **long**: pode dobrar o espaço de memória usado por um número (**int** ou **double**).

## Alguns tipos de dados modificados em C:

Tipo	Bytes	Escala
<b>unsigned char</b>	1	0 ... 255
<b>unsigned int</b>	4	0 ... 4294967295
<b>short int</b>	2	-32767 ... +32767
<b>long long int</b>	8	-9223372036854775808 ... +9223372036854775807
<b>unsigned short int</b>	2	0 ... 65535
<b>unsigned long long int</b>	8	0 ... 18446744073709551615

Os efeitos específicos dos modificadores **short** e **long** dependem do compilador usado!



# Introdução

## O operador `sizeof`

informa a quantidade de memória (em bytes) usada por um tipo de dados.

### Exemplo 3. Tamanho de um tipo

```
#include <stdio.h>
int main(void) {
    char v = 'A';
    printf("%zu\n", sizeof(v));
    printf("%zu\n", sizeof(int));
    printf("%zu\n", sizeof(v+1));
    printf("%zu\n", sizeof(1));
    printf("%zu\n", sizeof(1.2));
    printf("%zu\n", sizeof(double));
    return 0;
}
```

O especificador `%zu` indica que o valor que será exibido é dado por `sizeof!`



# Introdução

## O arquivo `stdio.h` (*standard input/output header*)

declara funções que lêem dados do teclado e exibem dados no vídeo.

### Observações:

- Há diversos dispositivos de entrada e saída de dados em um computador (e.g., mouse, impressora, disco, ...).
- O teclado é a **entrada de dados padrão** do computador.
- O vídeo é a **saída de dados padrão** do computador.
- A entrada de dados pode ser feita com a função **`scanf()`**.
- A saída de dados pode ser feita com a função **`printf()`**.
- Essas funções fazem entrada e saída de dados **formatados** (por isso seus nomes terminam com **f**).
- A formatação dos dados é definida por meio de **especificadores** de formato e **caracteres de controle**.



As funções **`printf()`** e **`scanf()`** são as mais usadas; porém, há várias outras!



# Introdução

## Especificador de formato

determina o tipo de dado que será lido do teclado ou exibido no vídeo.

Especificador	Representa
<code>%c</code>	caractere
<code>%o</code> , <code>%d</code> , <code>%x</code> , <code>%X</code>	número inteiro em octal, decimal ou hexadecimal
<code>%u</code>	número inteiro em base decimal sem sinal
<code>%hd</code>	número inteiro curto em base decimal
<code>%lld</code>	número inteiro longo longo em base decimal
<code>%f</code>	número real de precisão simples ou dupla
<code>%s</code>	cadeia de caracteres (string)
<code>%%</code>	único sinal de porcentagem

Esses são os especificadores mais usados; porém, há vários outros!



# Introdução

## Exemplo 4. Conversão de base decimal para base hexadecimal

Dado um número inteiro em base decimal, exibir o valor correspondente em base hexadecimal.

```
// Converte decimal em hexadecimal
#include <stdio.h>
int main(void) {
    int n;
    printf("Decimal? ");
    scanf("%d", &n);
    printf("Hexadecimal= %x\n", n);
    return 0;
}
```

## Exercício 3. Conversão de base hexadecimal para base decimal

Altere o programa do exemplo anterior para que a conversão seja de hexadecimal para decimal.



# Introdução

## Caractere de controle

representa um caractere especial da tabela ASCII.

Caractere de controle	Efeito na saída de dados
\a	soa o alarme do microcomputador
\b	o cursor retrocede à coluna anterior
\f	alimenta página na impressora
\n	o cursor avança para uma nova linha
\r	o cursor retrocede para a primeira coluna da linha
\t	o cursor avança para próxima marca de tabulação
\"	exibe uma única aspa
\'	exibe um único apóstrofo
\\\	exibe uma única barra invertida
\0	indica o final de uma cadeia de caracteres

Esses são os caracteres de controle mais usados; porém, há vários outros!



## Introdução

### Exemplo 5. Efeitos dos caracteres de controle

Execute o programa a seguir e analise o resultado.

```
#include <stdio.h>

int main(void) {
    printf("c:\backup\texto\novo\arq.txt\n");
    return 0;
}
```

### Exercício 4. Efeitos dos caracteres de controle

Altere o programa do exemplo anterior para que a saída exibida em vídeo seja:

**c :\backup\texto\novo\arq.txt**



# Introdução

## A função printf ()

exibe dados no vídeo do computador (saída padrão).

```
printf("formatação", val1, val2, ..., valn);
```

valores

- especificadores de formato (%): são substituídos pelos valores correspondentes.
- caracteres de controle (\): produzem efeitos especiais (e.g., som e mudança de linha).
- demais caracteres : são exibidos literalmente no vídeo do computador.

## Exemplo 6. Saída formatada

```
int idade;  
char sexo;  
...  
printf("%d %c\n", idade, sexo);  
...
```



# Introdução

## Formatação de campos para exibição

- Preenchimento com espaços...: % *tamanho\_do\_campo* **d**
- Preenchimento com zeros.....: %0 *tamanho\_do\_campo* **d**
- Número de casas decimais.....: % *tamanho\_do\_campo* . *número\_de\_casas* **f**

### Exemplo 7. Formatização de campos

```
#include <stdio.h>
int main(void) {
    int a = 678;
    float b = 12.3416;
    printf("%5d\n",a);           // 678
    printf("%06d\n",a);         // 000678
    printf("%7.3f\n",b);        // 12.342
    return 0;
}
```



# Introdução

## A função `scanf()`

lê dados do teclado do computador (**entrada padrão**) e os armazena em variáveis.

```
scanf ("formatação", end1, end2, ..., endn);
```

endereços de variáveis

- especificadores de formato (%): indicam os tipos e a ordem em que os dados devem ser digitados.
- demais caracteres : devem ser digitados pelo usuário.

## Exemplo 8. Entrada formatada

```
int idade;  
char sexo;  
...  
scanf ("%d %c", &idade, &sexo);  
...
```



# Introdução

## Exercício 5. Código ASCII

Dado um caractere, exiba seu código ASCII em octal, decimal e hexa.

```
#include <stdio.h>

int main(void) {
    char c;
    printf("Caractere? ");
    scanf("%c", &c);
    printf("ASCII em octal = %o\n", c);
    printf("ASCII em decimal = %d\n", c);
    printf("ASCII em hexadecimal = %X\n", c);
    return 0;
}
```

## Exercício 6. Código ASCII

[2<sup>a</sup> versão]

Altere o programa para ler um código ASCII em decimal e exibir o caractere correspondente.



# Introdução

## Operadores aritméticos

são usados para representar operações aritméticas:

- **Soma**.....: +
  - **Subtração**...: -
  - **Produto** ....: \*
  - **Divisão**.....: /
  - **Resto**.....: %
- } (o resultado é *inteiro* apenas quando os *dois* valores são *inteiros*)
- } (produz o resto da divisão entre *dois* números *inteiros*)

## Exemplo 9. Resultado dos operadores aritméticos

- 1 + 2       $\Rightarrow$  3
- 8.0 - 4      $\Rightarrow$  4.0
- 3 \* 7        $\Rightarrow$  21
- 7 / 2        $\Rightarrow$  3
- 7 / 2.0       $\Rightarrow$  3.5
- 7 % 2        $\Rightarrow$  1
- 7.0 % 2      $\Rightarrow$  erro



## Introdução

### Exercício 7. Média

Dadas as duas notas de prova de um aluno, informe a sua média.

### Exercício 8. Consumo

Dada uma distância percorrida em quilômetros, e o total de litros de combustível gasto para percorrê-la, informe o consumo médio do veículo.

### Exercício 9. Temperatura

Dada uma temperatura em graus Fahrenheit ( $^{\circ}\text{F}$ ), informe a correspondente em graus Celsius ( $^{\circ}\text{C}$ ). [Dica:  $\mathbf{C} = (\text{F}-32) * (5/9)$ ].

### Exercício 10. Distância entre dois pontos

Dadas as coordenadas ( $x_p, y_p$ ) e ( $x_q, y_q$ ), de dois pontos **P** e **Q** no plano cartesiano, exiba a distância entre eles. [Dica: use o Teorema de Pitágoras].

**Fim**

