

# Estrutura Sequencial

Prof<sup>a</sup> Ma. Luciane Y. H. Kanashiro

# Estrutura Sequencial

*A Estrutura sequencial de um algoritmo corresponde ao fato de que o conjunto de ações primitivas será executado em um **sequência linear** de cima para baixo e da esquerda para direita.*

FORBELLONE &EBERSPÄCHER

Estrutura Sequencial

**C**

# Estrutura sequencial

- Case sensitive

```
#include <nome_da_biblioteca>
void main()
{
    bloco de comandos;
}
```

# Escopo

- Blocos:
  - delimitados por { }
  - definem o escopo de uma entidade
- Métodos de acesso
  - public: método acessível em outros programas
  - private: método acessível apenas no programa em que foi definido
  - protected: método acessível na classe, subclasses

# Declaração de variáveis

- Deve ser declarada antes de ser utilizada
- Após especificação de **tipos**
- Exemplos:
  - float x;
  - int y, z;
  - char sexo;
- Reserva memória para armazenar seus valores

# Declaração de variáveis

- Boa prática:
  - Definir nomes que lembrem a finalidade
  - Nomes mais longos podem tornar o programa mais claro

# Tipos

A um tipo de dados está associado:

- uma representação
- o tamanho de células de memória para armazenar esta representação



| TIPO          | FAIXA DE VALORES                                 | TAMANHO (aproximado) |
|---------------|--|----------------------|
| char          | -128 a 127                                       | 8 bits               |
| unsigned char | 0 a 255  | 8 bits               |
| int           | -32.768 a 32.767                                 | 16 bits              |
| unsigned int  | 0 a 65.535                                       | 16 bits              |
| short int     | -32.768 a 32.767                                 | 16 bits              |
| long          | -2.147.483.648 a 2.147.483.647                   | 32 bits              |
| unsigned long | 0 a 4.294.967.295                                | 32 bits              |
| float         | $3.4 \times 10^{-38}$ a $3.4 \times 10^{38}$     | 32 bits              |
| double        | $1.7 \times 10^{-308}$ a $1.7 \times 10^{308}$   | 64 bits              |
| long double   | $3.4 \times 10^{-4932}$ a $1.1 \times 10^{4932}$ | 80 bits              |



# Tipos e Declaração de variáveis

## EXERCÍCIO DE FIXAÇÃO I

- 1.1 Determine qual é o tipo primitivo de informação presente nas sentenças a seguir:
- a) A placa “*Pare!*” tinha 2 furos de bala.
  - b) Josefina subiu 5 degraus para pegar uma maçã boa.
  - c) Alberta levou 3,5 horas para chegar ao hospital onde concebeu uma garota.
  - d) Astrogilda pintou em sua camisa: “*Preserve o meio ambiente*”, e ficou devendo \$ 100,59 ao vendedor de tintas.
  - e) Felisberto recebeu sua 18ª medalha por ter alcançado a marca de 57,3 segundos nos 100 metros rasos.

# Declaração de Constantes

- Deve ser declaradas antes do método main.
- Sintaxe:

```
#define <nome_da_constante> <valor>
```

Exemplo:

```
#define x = 8;
```

```
#define nome = "MARIA";
```

# Comando de atribuição

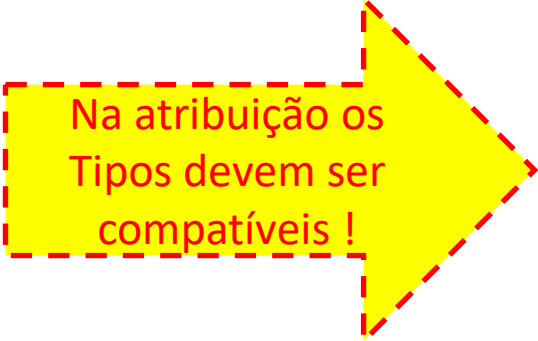
- Concede valores ou operações a variáveis
- Exemplos:

`x = 4;`

`x = x + 2;`

`y = 2.5;`

`sexo = 'f';`



Na atribuição os  
Tipos devem ser  
compatíveis !

`a = b + c;`

# Comando de entrada

- Recebe dados digitados
- Exemplos:

```
printf("digite um numero: ");  
scanf("%d",&numero);
```

# Comando de saída

- Usado para mostrar dados na tela

```
printf("%d",Y);
```

```
printf("Conteúdo de Y = %d",Y);
```

# Comentários

- São textos que podem ser inseridos em programas como o objetivo de documentá-los.

```
/*  
linhas de comentário  
linhas de comentário  
*/
```

```
// comentário
```

# Códigos Especiais

| Códigos especiais | Significado                                    |
|-------------------|--|
| \n                | Nova linha.                                    |
| \t                | Tabulação.                                     |
| \b                | Retrocesso (usado para impressora).            |
| \f                | Salto de página de formulário.                 |
| \a                | Beep – Toque do auto-falante.                  |
| \r                | CR – Retorno do cursor para o início da linha. |
| \\                | \ – Barra invertida.                           |
| \0                | Zero.  |
| \'                | Aspas simples (apóstrofo).                     |
| \"                | Aspas dupla.                                   |
| \xdd              | Representação hexadecimal.                     |
| \ddd              | Representação octal.                           |

# Códigos Especiais

| Códigos de formatação para printf() | Significado  |
|-------------------------------------|--|
| %c                                  | Caractere simples.                                 |
| %d                                  | Inteiro decimal com sinal.                         |
| %i                                  | Inteiro decimal com sinal.                         |
| %e                                  | Notação científica (e minúsculo).                  |
| %E                                  | Notação científica (E maiúsculo).                  |
| %f                                  | Ponto flutuante em decimal.                        |
| %g                                  | Usa %e ou %f, o que for menor.                     |
| %G                                  | Usa %E ou %f, o que for menor.                     |
| %o                                  | Inteiro octal sem sinal.                           |
| %s                                  | String de caracteres.                              |
| %u                                  | Inteiro decimal sem sinal.                         |
| %x                                  | Inteiro hexadecimal sem sinal (letras minúsculas). |
| %X                                  | Inteiro hexadecimal sem sinal (letras maiúsculas). |
| %p                                  | Ponteiro (endereço).                               |
| %n                                  | Ponteiro inteiro.                                  |
| %%                                  | Imprime um caractere %.                            |



# Operadores

- Operadores aritméticos

| OPERADOR | EXEMPLO | COMENTÁRIO   |
|----------|---------|--|
| =        | $x = y$ | O conteúdo da variável Y é atribuído à variável X. (A uma variável pode ser atribuído o conteúdo de outra, um valor constante ou, ainda, o resultado de uma função.) |
| +        | $x + y$ | Soma o conteúdo de X e de Y.   |
| -        | $x - y$ | Subtrai o conteúdo de Y do conteúdo de X.  |
| *        | $x * y$ | Multiplica o conteúdo de X pelo conteúdo de Y.   |

continua

| OPERADOR | EXEMPLO  | COMENTÁRIO                               |
|----------|----------|--|
| /        | $x / y$  | Obtém o quociente da divisão de X por Y. |
| %        | $x \% y$ | Obtém o resto da divisão de X por Y.     |

# Operadores

- Prioridades:

pot   rad   sqr

\*   /   div   mod

+   -

# Operadores

- Operadores aritméticos

| OPERADOR        | EXEMPLO              | COMENTÁRIO  |
|-----------------|----------------------|---|
| <code>+=</code> | <code>x += y</code>  | Equivale a <code>X = X + Y</code> .                             |
| <code>-=</code> | <code>x -= y</code>  | Equivale a <code>X = X - Y</code> .                             |
| <code>*=</code> | <code>x *= y</code>  | Equivale a <code>X = X * Y</code> .                             |
| <code>/=</code> | <code>x /= y</code>  | Equivale a <code>X = X / Y</code> .                             |
| <code>%=</code> | <code>x %= y</code>  | Equivale a <code>X = X % Y</code> .                             |
| <code>++</code> | <code>x++</code>     | Equivale a <code>X = X + 1</code> .                             |
| <code>++</code> | <code>y = ++x</code> | Equivale a <code>X = X + 1</code> e depois <code>Y = X</code> . |
| <code>++</code> | <code>y = x++</code> | Equivale a <code>Y = X</code> e depois <code>X = X + 1</code> . |
| <code>--</code> | <code>x--</code>     | Equivale a <code>X = X - 1</code> .                             |
| <code>--</code> | <code>y = --x</code> | Equivale a <code>X = X - 1</code> e depois <code>Y = X</code> . |
| <code>--</code> | <code>y = x--</code> | Equivale a <code>Y = X</code> e depois <code>X = X - 1</code> . |

# Operadores

- Operadores relacionais

| OPERADOR | EXEMPLO                | COMENTÁRIO   |
|----------|------------------------|--|
| ==       | <code>x == y</code>    | O conteúdo de X é igual ao conteúdo de Y.          |
| !=       | <code>x != y</code>    | O conteúdo de X é diferente do conteúdo de Y.      |
| <=       | <code>x &lt;= y</code> | O conteúdo de X é menor ou igual ao conteúdo de Y. |
| >=       | <code>x &gt;= y</code> | O conteúdo de X é maior ou igual ao conteúdo de Y. |
| <        | <code>x &lt; y</code>  | O conteúdo de X é menor que o conteúdo de Y.       |
| >        | <code>x &gt; y</code>  | O conteúdo de X é maior que o conteúdo de Y.       |

# Palavras reservadas

**PALAVRAS RESERVADAS DE C/C++** são nomes utilizados pelo compilador para representar comandos de controle do programa, operadores e diretivas.

|          |           |            |            |           |          |
|----------|-----------|------------|------------|-----------|----------|
| asm      | _asm      | _asm       | auto       | break     | case     |
| cdecl    | _cdecl    | _cdecl     | char       | class     | const    |
| continue | _cs       | _cs        | default    | delete    | do       |
| double   | _ds       | _ds        | else       | enum      | _es      |
| _es      | _export   | _export    | extern     | far       | _far     |
| _far     | _fastcall | _fastcall  | float      | for       | friend   |
| goto     | huge      | _huge      | _huge      | if        | inline   |
| int      | interrupt | _interrupt | _interrupt | _loadds   | _loadds  |
| long     | near      | _near      | _near      | new       | operator |
| pascal   | _pascal   | _pascal    | private    | protected | public   |
| register | return    | _saveregs  | _saveregs  | _seg      | _seg     |
| short    | signed    | sizeof     | _ss        | _ss       | static   |
| struct   | switch    | template   | this       | typedef   | union    |
| unsigned | virtual   | void       | volatile   | while     |          |

# Exemplos

- Faça um programa que receba 2 números e mostre a soma dos mesmos.

# Exemplos

- Faça um programa que receba 3 notas , calcule e mostre a média aritmética

# Exemplos

1. Faça um algoritmo para “Calcular o estoque médio de uma peça”, sendo que

$$\text{ESTOQUE M\u00c9DIO} = (\text{QUANTIDADE M\u00cdNIMA} + \text{QUANTIDADE M\u00c1XIMA})$$

3. Teste o algoritmo anterior com dados definidos por voc\u00ea.



# Links

- Dev c++
- [https://sourceforge.net/projects/orwelldevcpp/?source=typ\\_redirect](https://sourceforge.net/projects/orwelldevcpp/?source=typ_redirect)

# Referências

ASCENCIO, A. CAMPOS, E. **Fundamentos da programação de computadores: algoritmos, Pascal, C/C++ e Java.** 1 ed. São Paulo: Pearson Prentice Hall, 2007.

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. **Lógica de programação: a construção de algoritmos e estrutura de dados.** 2. ed. São Paulo: Makron Books, 2000. 197 p.cap