



FeMASS

FACULDADE PROF. MIGUEL ÂNGELO DA SILVA SANTOS

Estrutura de um Programa C

Ana Maria M. Moura

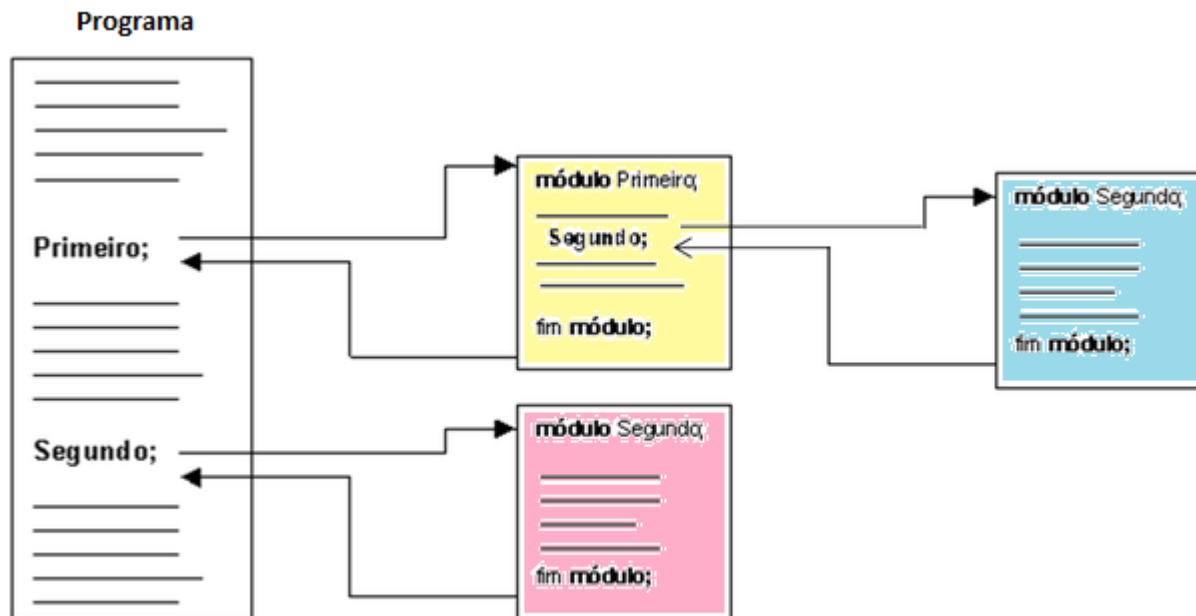
professora.anamoura@gmail.com

O que é um programa estruturado?

- Um programa representa de forma estruturada, um padrão de comportamento de eventos ou sequência de ações, que levam a um resultado esperado
- É uma sequência ordenada e finita de etapas (ou comandos), cuja execução, passo a passo, resolve um determinado problema.

O que é um programa estruturado?

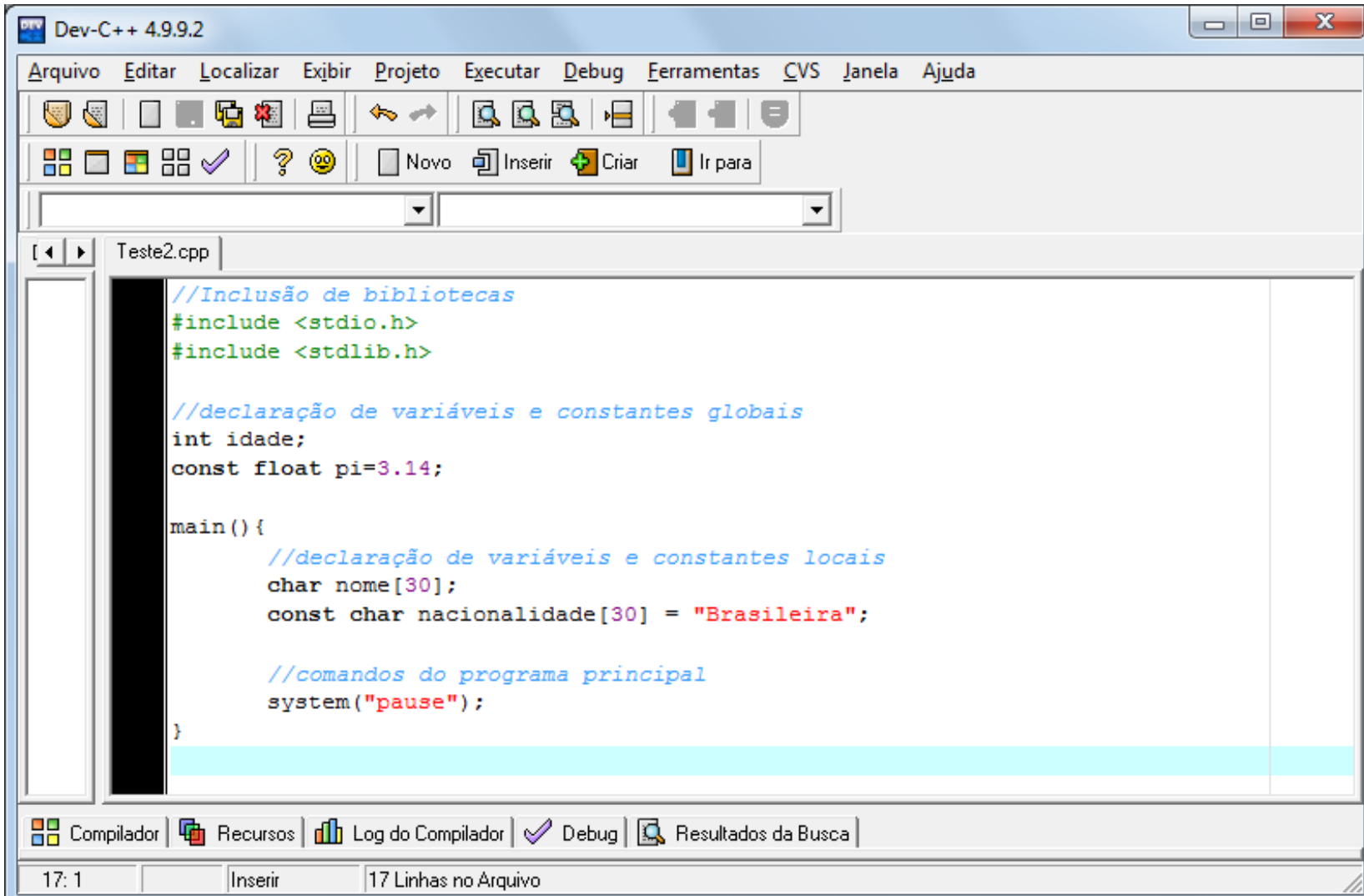
- Um programa estruturado pode ser modularizado. O C implementa a modularização sempre em funções, que podem ser Internas ou externas



Estrutura de um Programa C

- Todo programa em C é baseado em função Até o programa principal (chamado **main**) é uma função
- Veremos mais adiante no curso que toda função pode retornar um valor em seu nome
 - Por enquanto aceite isto, mas no decorrer do curso as funções serão detalhadas
- O retorno do valor da função deve, então, ser de um tipo declarado durante a codificação
 - Inteiro, caractere, real, etc...

Estrutura de um Programa C



The screenshot shows the Dev-C++ 4.9.9.2 IDE interface. The menu bar includes Arquivo, Editar, Localizar, Exibir, Projeto, Executar, Debug, Ferramentas, CVS, Janela, and Ajuda. The toolbar contains various icons for file operations, editing, and execution. The main editor window displays the code for 'Teste2.cpp'.

```
//Inclusão de bibliotecas
#include <stdio.h>
#include <stdlib.h>

//declaração de variáveis e constantes globais
int idade;
const float pi=3.14;

main() {
    //declaração de variáveis e constantes locais
    char nome[30];
    const char nacionalidade[30] = "Brasileira";

    //comandos do programa principal
    system("pause");
}
```

The status bar at the bottom shows the current line and column (17: 1), the current operation (Inserir), and the total number of lines in the file (17 Linhas no Arquivo).



FeMASS

FACULDADE PROF. MIGUEL ÂNGELO DA SILVA SANTOS

Estrutura de um Programa C

```
Dev-C++ 4.9.9.2
Arquivo  Editar  Localizar  Exibir  Projeto  Executar  Debug  Ferramentas  CVS  Janela  Ajuda

//Inclusão de bibliotecas
#include <stdio.h>
#include <stdlib.h>

//declaração de variáveis e constantes globais
float resultado;

float soma(float operando1, float operando2){
    return(operando1 + operando2);
}

main(){
    //declaração de variáveis e constantes locais
    float op1, op2;
    printf("Informe dois números: ");
    scanf("%f %f", &op1, &op2);
    resultado = soma(op1,op2);
    printf("a soma resulta em %.2f ", resultado);
    //comandos do programa principal
    system("pause");
}
```

7: 1 Modificac Inserir 22 Linhas no Arquivo

Editor de Programas

- Nesta disciplina, utilizaremos o editor de programas Dev C++.



- Ele é gratuito e você pode obtê-lo em alguns sites de downloads.



Editor de Programas

Dev-C++ 4.9.9.2

Arquivo Editar Localizar Exibir Projeto Executar Debug Ferramentas CVS Janela Ajuda

Novo Inserir Criar Ir para

[<] [>] *] Teste2.cpp

```
//Inclusão de bibliotecas
#include <stdio.h>
#include <stdlib.h>

//declaração de variáveis e constantes globais
float resultado;

float soma(float operando1, float operando2){
    return(operando1 + operando2);
}

main()
{
    //declaração de variáveis e constantes locais
    float op1, op2;
    printf("Informe dois números: ");
    scanf("%f %f", &op1, &op2);
    resultado = soma(op1,op2);
    printf("a soma resulta em %.2f ", resultado);
    //comandos do programa principal
    system("pause");
}
```

Barra de Menu

Salva o arquivo.

Cria um novo arquivo fonte para inserção de código.

Compilar e Executar

Executar

Compilar

Editor de Programas

- **Arquivo:** possui as funções básicas de manuseio de arquivos (criar novo arquivo, abrir arquivo, fechar, imprimir, ver propriedades)
- **Editar:** aonde estão localizadas as funções de edição básicas de edição (copiar, recortar, colar) e algumas funções úteis para programação (como comentar e descomentar trechos do programa, e criar e acessar “bookmarks”, que são marcas de acesso rápido para partes do programa, especialmente úteis para programas extensos)
- **Localizar:** possui os comandos de procurar e substituir partes do código; o menu Exibir permite o controle de quais componentes da tela são exibidos
- **Projeto:** refere-se a projetos de programas que possuem vários componentes e arquivos de códigos separados e é utilizado para adicionar e retirar componentes do projeto
- **Executa:** é talvez o mais importante para nós, e nele estão localizadas as funções básicas do compilador (como os comandos Compilar, Executar) e algumas funções úteis como procurar por erros de sintaxe

Editor de Programas

- **Debug:** serve para controlar o debug de um programa, que é a sua execução passo-a-passo para melhor análise e busca por erros
- **Ferramentas:** refere-se a várias opções do compilador, do ambiente de trabalho e de edição, além de configurações diversas
- **CVS:** é uma função extra do compilador, e não nos tem serventia
- **Janela:** possui comandos úteis para os casos em que temos vários arquivos ou projetos abertos ao mesmo tempo e precisamos alternar entre eles.
- **Ajuda:** dá acesso à ajuda do programa, que possui uma listagem dos principais comandos do compilador e um breve tutorial da linguagem C.

Comentários

- Não é obrigatório
 - Entretanto é interessante que todo programa o tenha logo no início
- Deve conter o nome do programa e uma explicação da proposta do programa
- Pode conter também o nome do programador, data de criação, data da última manutenção, etc...
- Se for comentário de apenas uma linha
// Escreva nesta linha o comentário
- Se for comentado em mais de uma linha
/* Escreva nestas linhas o comentário
Escreva nestas linhas o comentário
Escreva nestas linhas o comentário ***/**
- **Todo comentário é ignorado pelo compilador**

Bibliotecas

- Bibliotecas de funções C
- O C ANSI nomeia e descreve um conjunto de funções que todo compilador padrão deve suportar
- Estas funções estão em arquivos cabeçalho(headers), que são conhecidos como biblioteca de funções C
- São arquivos com extensão “*.h” (header) e ficam no diretório:
 - ...\\Dev-CppPortable\\App\\devcpp\\include

Bibliotecas

- Entre as bibliotecas existentes, destacam-se:

Arquivo de cabeçalho

ASSERT.H
CTYPE.H
ERRNO.H
FLOAT.H
LIMITS.H
LOCALE.H
MATH.H
SETJMP.H
SIGNAL.H
STDARG.H
STDDEF.H
STDIO.H
STDLIB.H
STRING.H
TIME.H

Finalidades

Define a macro `assert()`
Manipulação de caracteres
Apresentação de erros
Define valores em ponto flutuante dependentes da implementação
Define limites em ponto flutuante dependentes da implementação
Suporta localização
Diversas definições usadas pela biblioteca de matemática
Suporta desvios não-locais
Suporta manipulação de sinal
Suporta listas de argumentos de comprimento variável
Define algumas constantes normalmente usadas
Suporta E/S com arquivos
Declarações miscelâneas
Suporta funções de strings
Suporta as funções de horário do sistema

Bibliotecas

- Para incluir uma biblioteca, devemos utilizar:
 - `#include <nome_biblioteca.h>`

Ex.: `#include <stdio.h>`

Delimitadores de Bloco

- Dizemos que temos um bloco de comandos quando temos:

Um delimitador de inicialização “{”

Vários comandos

Um delimitador de finalização “}”

Exemplo:

{

Comando 1

Comando 2

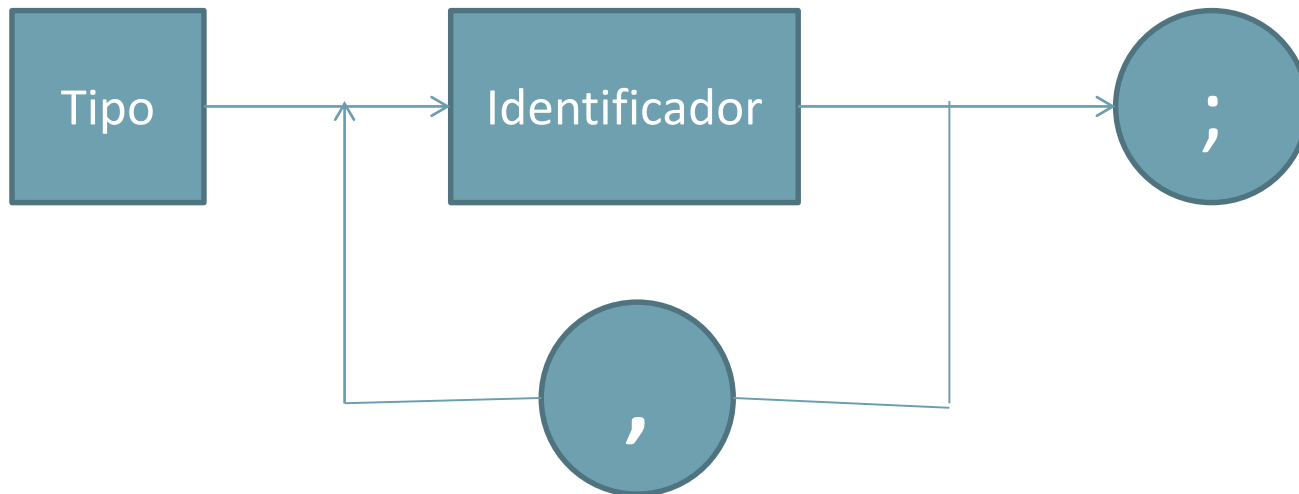
Comando 3

...

}

Declaração de Variáveis e Constantes

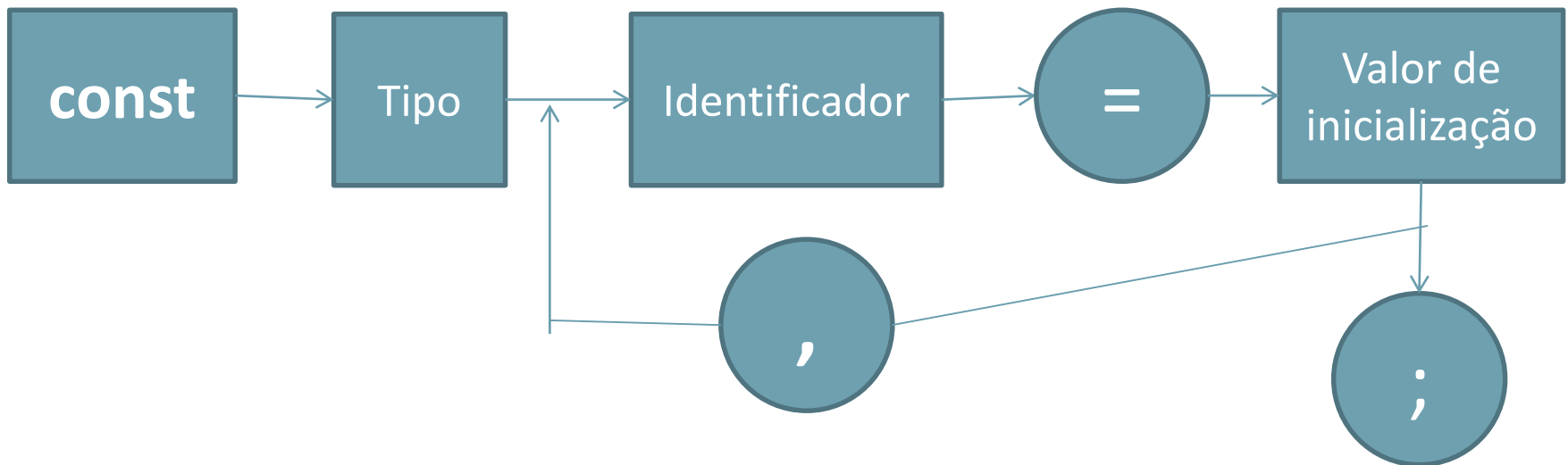
- Declaração de Variáveis



```
float operando1, operando2;
```


Declaração de Variáveis e Constantes

- Declaração de Constantes

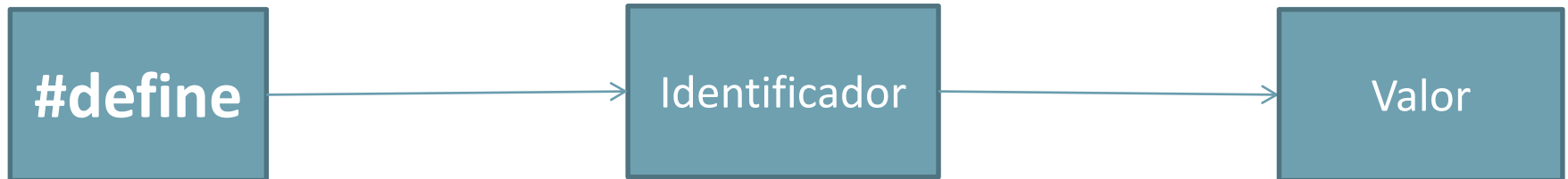


```
const float pi=3.14;
```

É uma declaração de identificador presente no código com escopo e tempo de vida no código. É verificado pelo compilador e tem um tipo. Pode ser usado em todas as circunstâncias onde seria usado uma variável.

Declaração de Variáveis e Constantes

- Declaração de Constantes



```
#define pi 3.14; /* EVITE*/
```

É uma diretiva de preprocessador de código. É apenas um texto sendo substituído por outro sem nenhum tipo de verificação.

Declaração de Variáveis e Constantes

- Escopo de variáveis
- **Globais**
 - são aquelas declaradas fora do escopo das funções;
 - Podem ser acessadas a partir de qualquer lugar do programa;
- **Locais**
 - são aquelas declaradas no início de um bloco e seu escopos estão restritos aos blocos em que foram declaradas.



FeMASS

FACULDADE PROF. MIGUEL ÂNGELO DA SILVA SANTOS

Declaração de Variáveis e Constantes

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

#define IMC_MINIMO 18.5 /*constante global sem verificação de tipo - evite*/
const float IMC_NORMAL=24.9; /*constante global com verificação de tipo*/
float imc=0; /*variável global*/

main(){
    #define IMC_SOBREPESO 29.9 /*constante local sem verificação de tipo - evite*/
    const float IMC_OBESIDADE_I=34.9; /*constante local com verificação de tipo*/
    const float IMC_OBESIDADE_II=39.9; /*constante local com verificação de tipo*/
    float peso, altura; /*variável local*/
    setlocale(LC_ALL,"portuguese");
    printf("Peso: ");
    scanf("%f",&peso);
    printf("Altura: ");
    scanf("%f",&altura);
    imc=peso/(altura*altura);
    printf("IMC= %.2f (%s) \n",imc,
    imc<IMC_MINIMO?"Subnutrido":
    imc<=IMC_NORMAL?"Normal":
    imc<=IMC_SOBREPESO?"Sobrepeso":
    imc<=IMC_OBESIDADE_I?"Obeso":
    imc<=IMC_OBESIDADE_II?"Obesidade grau II":"Obesidade mórbida");
    system("pause");
}
```