Modularização

Prof. Manassés Ribeiro

Agenda

1. Decomposição de problemas
2. Passos para a decomposição de problemas
3. Tipos de módulos
    a. Módulos de ação (procedimentos)
    b. Módulos de retorno (funções)
4. Passagem de parâmetros
5. Exemplos de implementação em C

Decomposição de problemas

- Fator determinante para a **redução da complexidade**;
- Ajuda a **simplificar a resolução**;
- Permite focar a **atenção em um problema pequeno** por vez;
- Facilita a **compreensão do todo**;
- **Processo de refinamentos sucessivos.**

Passos para a decomposição de problemas

- **Dividir** o problema em suas **partes principais**;
- **Analisar** a divisão obtida para **garantir coerência** e **coesão**;
  - Se alguma parte permanecer complexa, deve-se decompô-la ainda mais;
- **Verificar** o resultado para **garantir o entendimento, a coerência e a coesão**.

Tipos de módulos

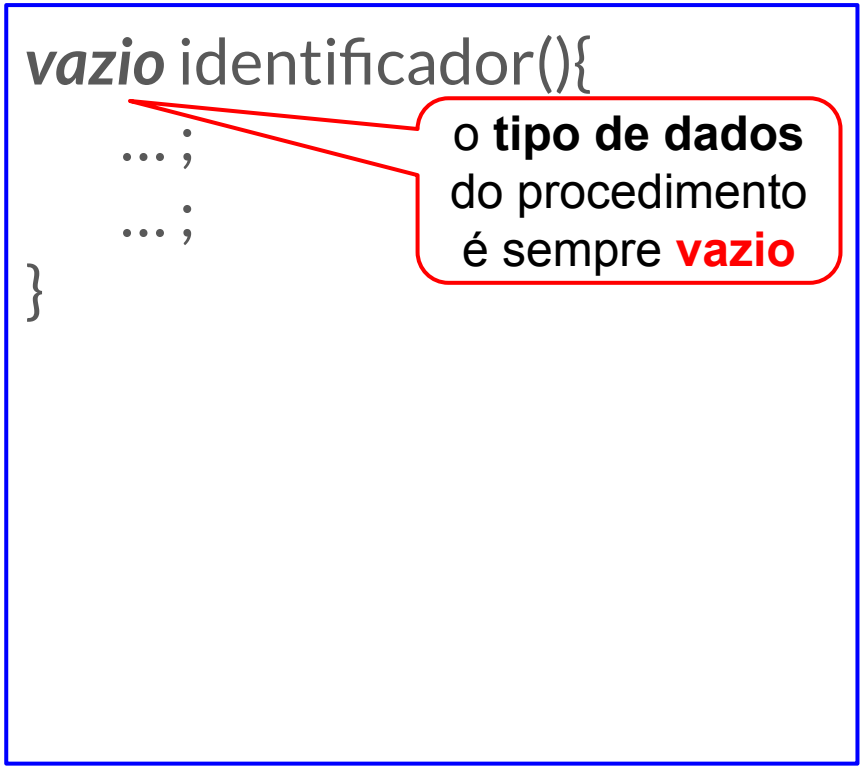
- Módulos de **ação**:
  - também conhecidos como **procedimentos**;
- Módulos de **resultado**:
  - também conhecidos como **funções**;
- Os módulos (funções e procedimentos) devem ser **declarados fora do programa principal (*main*())**.

Módulos de ação (Procedimentos)

- São estruturas que **agrupam um conjunto de comandos**, que são executados quando o procedimento é chamado;
- Todo procedimento possui um **tipo de dado** que **é vazio** (***void***);
- Para cada procedimento é definido um **identificador único**:
  - que seguem as mesmas regras dos identificadores das variáveis.

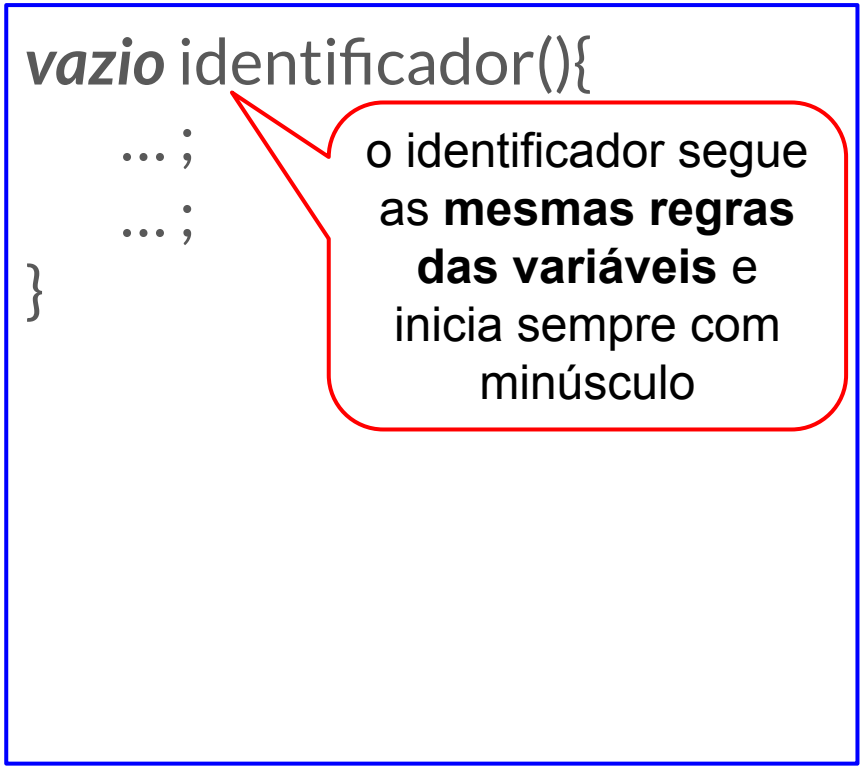
Módulos de ação (Procedimentos)

Declaração de procedimentos:



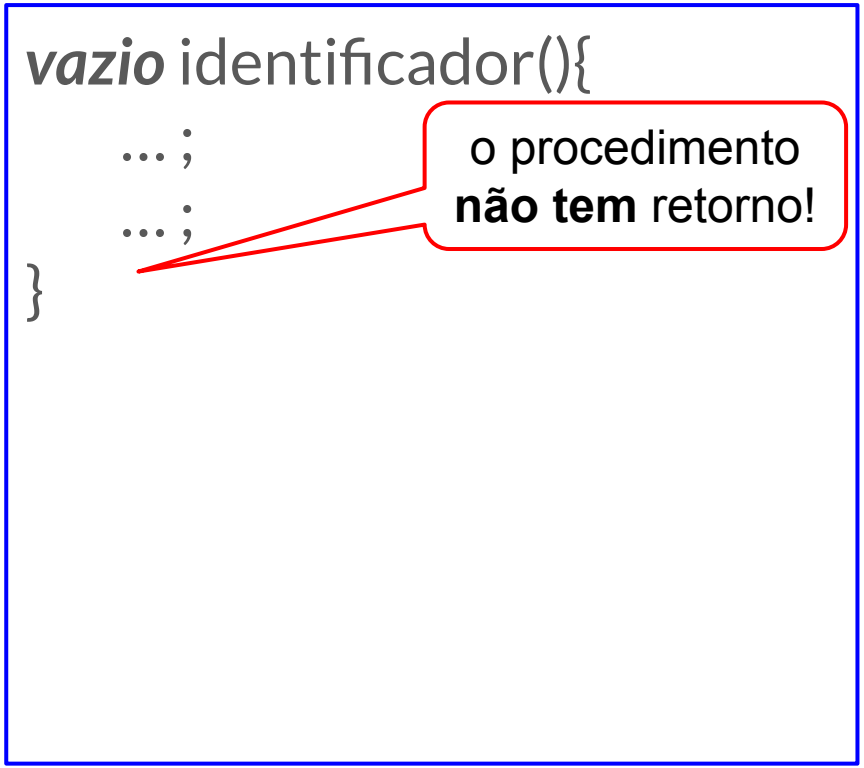
Módulos de ação (Procedimentos)

Declaração de procedimentos:



Módulos de ação (Procedimentos)

Declaração de procedimentos:



Módulos de ação (Procedimentos)

Exemplo em C:

```
void escreve_Olah(){
    printf("Olah mundo\n");
}

int main(){
    escreve_Olah();
}
```

Módulos de ação (Procedimentos)

Exemplo em C:

```
void escreve_Olah(){
    printf("Olah mundo\n");
}

int main(){
    escreve_Olah();
}
```

Neste ponto é executada a **chamada** do procedimento

Módulos de resultado (Funções)

- São módulos que **retornam um valor ao final de sua execução (único no caso do C)**;
- Toda função deve ter um **tipo de dado** o qual determina o **tipo de dado do retorno**;
- A expressão contida dentro do comando **return** é chamada de **valor de retorno** (é a resposta da função);
- O comando **return** é sempre executado por último.

Módulos de resultado (Funções)

Declaração de funções:

```
tipo_de_dados identificador(){
        ... ;
        ... ;
        ... ;
        return (....);
}
```

O **tipo de dados** da função identifica o tipo de dados do **retorno**

Módulos de resultado (Funções)

Exemplo em C:

```
int soma_a_b(){
        int a, b;
        printf("Digite o valor de a: ");
        scanf("%i", &a);
        printf("Digite o valor de b: ");
        scanf("%i", &b);
        return (a + b);
}

int main(){
        int c;
        c = soma_a_b();
        printf("O resultado da soma eh: %i\n", c);
}
```

Módulos de resultado (Funções)

Exemplo em C:

```
int soma_a_b(){
        int a, b;
        printf("Digite o valor de a: ");
        scanf("%i", &a);
        printf("Digite o valor de b: ");
        scanf("%i", &b);
        return (a + b);
}
int main(){
        int c;
        c = soma_a_b();
        printf("O resultado da soma eh: %i\n", c);
}
```

Neste ponto acontece a chamada da função **soma_a_b()**

Módulos de resultado (Funções)

Exemplo em C:

```
int soma_a_b(){
        int a, b;
        printf("Digite o valor de a: ");
        scanf("%i", &a);
        printf("Digite o valor de b: ");
        scanf("%i", &b);
        return (a + b);
}
int main(){
        int c;
        c = soma_a_b();
        printf("O resultado da soma eh: %i\n", c);
}
```

Aqui acontece o retorno da função **soma_a_b()** para quem a chamou (***main***)

Módulos de resultado (Funções)

Exemplo em C:

```
int soma_a_b(){
        int a, b;
        printf("Digite o valor de a: ");
        scanf("%i", &a);
        printf("Digite o valor de b: ");
        scanf("%i", &b);
        return (a + b);
}

int main(){
        int c;
        c = soma_a_b();
        printf("O resultado da soma eh: %i\n", c);
}
```

O retorno da função **soma_a_b()** é armazenado na variável "c"

Passagem de parâmetros

- Parâmetros são utilizados para **enviar informações para dentro do módulo**;
- Na prática, **os parâmetros** de um módulo **determinam seu comportamento**;
- Um módulo pode não ter parâmetros:
  - a ausência de parâmetros é mais comum em módulos de ação (procedimentos).

Passagem de parâmetros

Exemplo em C:

```
int soma_a_b(int v1, int v2){
        int s;
        s = v1 + v2;
        return s;
}
int main(){
        int a, b, r;
        printf("Digite o valor de a: ");
        scanf("%i", &a);
        printf("Digite o valor de b: ");
        scanf("%i", &b);

        r= soma_a_b(a, b);
        printf("O resultado da soma eh: %i\n", r);
}
```

Passagem de parâmetros

Exemplo em C:

```
int soma_a_b(int v1, int v2){
        int s;
        s = v1 + v2;
        return s;
}
int main(){
        int a, b, r;
        printf("Digite o valor de a: ");
        scanf("%i", &a);
        printf("Digite o valor de b: ");
        scanf("%i", &b);

        r= soma_a_b(a, b);
        printf("O resultado da soma eh: %i\n", r);
}
```

Parâmetros do módulo: neste caso são dois parâmetros do tipo inteiro (**int**)

Passagem de parâmetros

Exemplo em C:

```
int soma_a_b(int v1, int v2){
        int s;
        s = v1 + v2;
        return s;
}
int main(){
        int a, b, r;
        printf("Digite o valor de a: ");
        scanf("%i", &a);
        printf("Digite o valor de b: ");
        scanf("%i", &b);

        r= soma_a_b(a, b);
        printf("O resultado da soma eh: %i\n", r);
}
```

Quando é executada a chamada do módulo é necessário informar os parâmetros

Exercício

Construa um programa em C modularizado que calcule a média aritmética para 50 alunos considerando quatro notas bimestrais (n1, n2, n3, n4) e avalie se cada aluno foi aprovado (média >= 7) ou reprovado (média < 7).

Parâmetros: outros aspectos a serem considerados

- **Nunca** use variáveis globais (**heresia!**);
- Os parâmetros são de **dois tipos**: **valor** e **referência**;
- Passagem de parâmetros **por valores são os mais comuns** (visto no exemplo anterior);
- Passagem de parâmetros por referência será melhor explorado no **conteúdo de ADM**:
  - exemplo: passagem de **parâmetros de vetores** em C

Exercícios

1. Escreva um programa em C (modularizado) que leia um vetor de tamanho N (onde N é uma constante de tamanho 12), escreva o vetor e a soma de todos os elementos de índice par.
2. Escreva um programa em C (modularizado) para ler e armazenar valores inteiros em uma matriz (5,5). A seguir, calcular a média dos valores pares contidos na matriz e escrever a média calculada e o conteúdo da matriz.

Nesta aula foi visto

1. Decomposição de problemas
2. Passos para a decomposição de problemas
3. Tipos de módulos
    a. Módulos de ação (procedimentos)
    b. Módulos de retorno (funções)
4. Passagem de parâmetros
5. Exemplos de implementação em C

Prototipação

to be continued soon!