5950257 - Programação de Computadores

Aula 08a - Array Unidimensional

Prof. Dr. Denis M. L. Martins
DCM | FFCLRP | USP

Objetivos de Aprendizagem

- Compreender o conceito de vetor (array) em C
- Declarar e inicializar arrays corretamente
- Acessar e modificar elementos de um vetor
- Utilizar estruturas de repetição com arrays
- Desenvolver algoritmos utilizando arrays

> Para os exercícios desta aula, você pode utilizar IDEs online como:
>
> https://www.onlinegdb.com/online_c_compiler
>
> https://www.programiz.com/c-programming/online-compiler/

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Problema

Considere um programa que calcula a média de notas de um aluno.
Para cada prova é necessário criar uma variável que armazena o valor da nota.

```
int notaProva1 = 8;
int notaProva2 = 7;
int notaProva3 = 9;
```

Problemas:

- Se precisar adicionar mais uma prova, precisa criar mais uma variável.
- Variáveis estão desconectadas umas das outras, embora compartilhem o mesmo conceito.

Precisamos de um contêiner lógico que possa manter **múltiplos** elementos do mesmo tipo, na mesma ordem, sem poluir o código.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Solução com Arrays (Vetores)

Em vez de usar variáveis simples separadas, vamos utilizar uma variável composta.

```
int notas[3] = {8, 7, 9};
```

Vantagens:

- Código mais organizado: Mais fácil de manter (adaptar, modificar).
- Facilidade de processamento: Sem gerenciamento manual.
- Uso eficiente de memória: Acesso Rápido

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Array (ou Vetor)

Arrays são coleções de elementos do **mesmo tipo de dado**, armazenados em posições de memória sequenciais (contíguas).

```
tipo nome[tamanho];
```

Exemplo:

```
int notas[3];
```

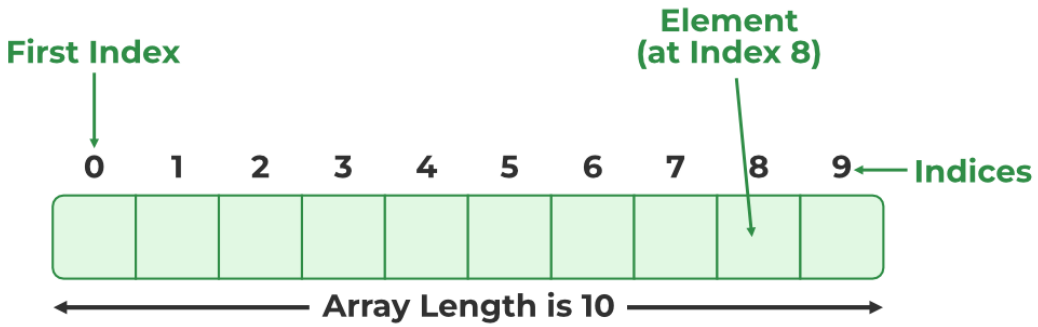
- **int**: tipo dos elementos
- **notas**: nome do vetor (variável)
- **[3]**: quantidade de elementos

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Acessando Elementos e Indexação Zero-Base

O acesso aos elementos é feito por meio de índices inteiros.

- A contagem de índices em C **sempre começa em zero (0)**.
- Em um array de tamanho N, os índices válidos são: $0, 1, 2, \ldots, N-1$.



> **Observação**: Em C, o tamanho de um array deve ser conhecido em tempo de compilação (compile-time constant).
>
> Fonte da imagem: GeeksForGeeks.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo

```c
#include <stdio.h>

int main() {

    int idade[3];

    idade[0] = 18;
    idade[1] = 20;
    idade[2] = 25;

    printf("%d\n", idade[0]);
    printf("%d\n", idade[1]);
    printf("%d\n", idade[2]);

    return 0;
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exercício

Crie um programa que:

1. Inicializa um vetor de tamanho 5 com valores: 1, 0, 0, 0, 2.
2. Imprime o primeiro e o último elementos.
3. Modifica o valor do elemento de índice 3 para 8.
4. Imprime o valor do elemento de índice 3.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Inicialização

Inicialização Direta:

```
int valores[5] = {1, 2, 3, 4, 5};
```

Inicialização Parcial:

```
int numeros[5] = {10, 20};
```

> Na inicialização parcial acima os demais valores são considerados zero: **10 20 0 0 0**

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo

```c
int numeros[4] = {10, 20, 30, 40}; // Tamanho 4. Índices de 0 a 3.

printf("Primeiro elemento (Índice 0): %d\n", numeros[0]); // Imprime 10
printf("Último elemento (Índice 3): %d\n", numeros[3]); // Imprime 40

// ATENÇÃO!
// printf("%d", numeros[4]); // ERRO: Índice fora dos limites (Out-of-Bounds)
```

> **Atenção**: Nunca acesse um índice que exceda o tamanho declarado do array. Isso leva a **Comportamento Indefinido (Undefined Behavior)**, corrompendo a memória e causando falhas imprevisíveis.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Inferência de Tamanho

```
int x[] = {1, 2, 3};
```

O compilador calcula automaticamente o tamanho.

> O compilador ele executa uma contagem estática dos valores literais fornecidos e ajusta o metadata do array para garantir que o bloco de memória seja grande o suficiente (e somente o necessário).

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Percorrendo Vetores com Loop

A maneira mais segura de processar todos os elementos é usando laços de repetição (**for** ou **while**).

```c
#include <stdio.h>
#define TAMANHO 5

int main() {
    int numeros[TAMANHO] = {10, 20, 30, 40, 50};
    for(int i = 0; i < TAMANHO; i++) {
        printf("%d\n", numeros[i]);
    }

    return 0;
}
```

> **Dica**: Usar constantes (**#define** ou **const int**) para definir o tamanho do array é uma prática essencial para evitar "números mágicos" (magic numbers).

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exercício

Modifique o programa abaixo para exibir os elementos do array **ordem reversa**:

```c
#include <stdio.h>
#define TAMANHO 5

int main() {
    int numeros[TAMANHO] = {10, 20, 30, 40, 50};
    for(int i = 0; i < TAMANHO; i++) {
        printf("%d\n", numeros[i]);
    }

    return 0;
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Leitura de Dados do Usuário

```c
#include <stdio.h>

int main() {

    int notas[4];

    for(int i = 0; i < 4; i++) {
        printf("Digite a nota: ");
        scanf("%d", &notas[i]);
    }

    return 0;
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exercício

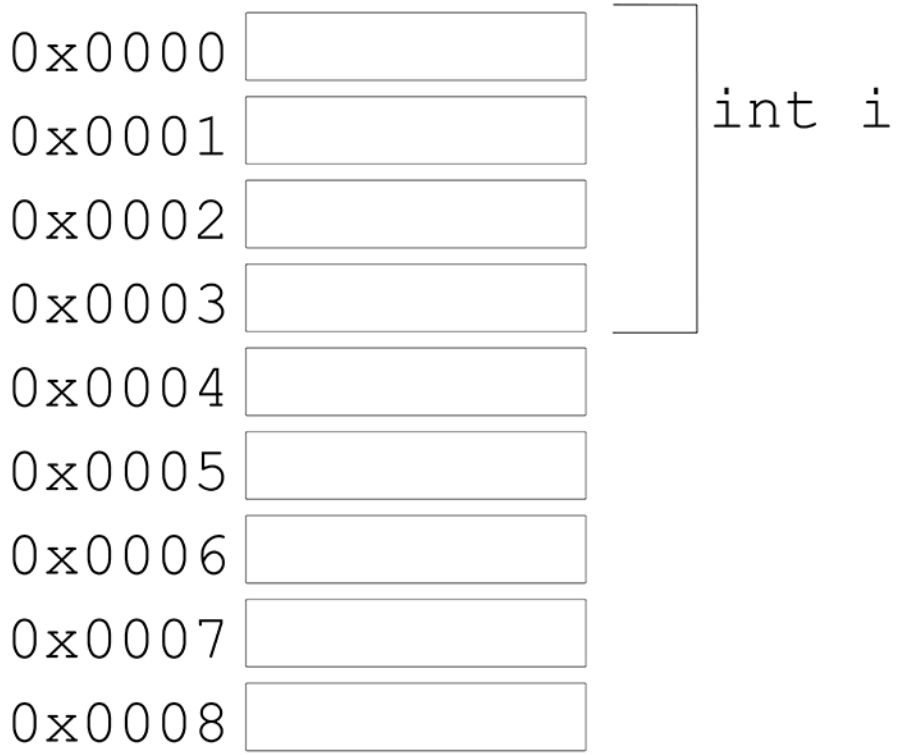
Crie um programa que:

- Leia 10 números inteiros e armazene em um vetor
- Calcule a Soma e a Média

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Lembrando: Variáveis em Memória

- Quando criamos uma variável, estamos alocando espaço na memória.
- Tipos diferentes de variável tomam espaços diferentes na memória. Exemplo:
  - **int**: 4 bytes
  - **double**: 8 bytes

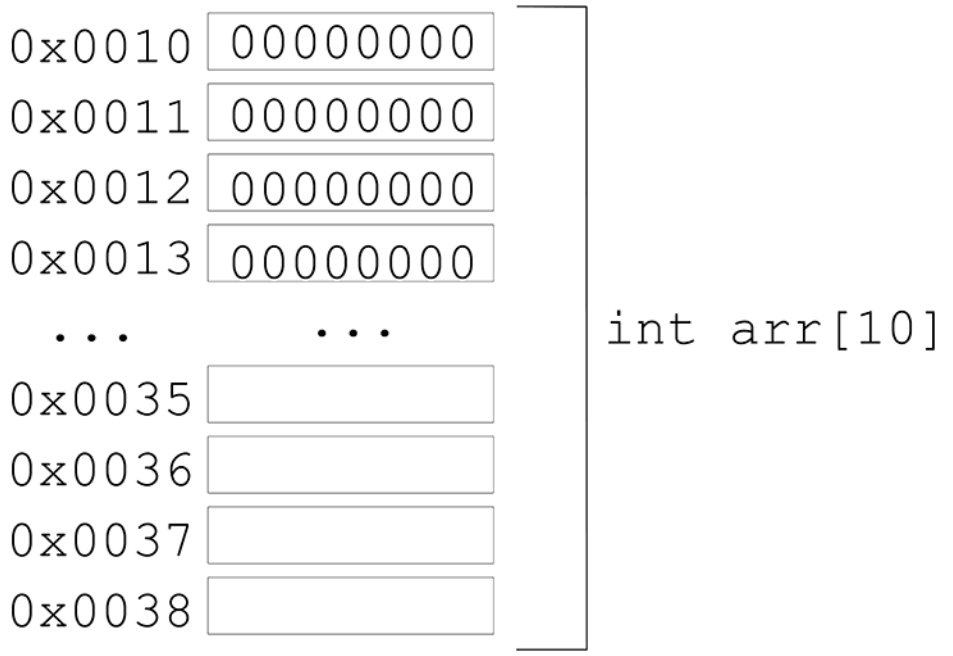


> Fonte da imagem: openframeworks.cc

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Array em Memória

- Um array, por definição, é um bloco de memória contíguo.
- Todos os elementos do array ocupam posições de endereços de memória adjacentes (um após o outro).
- Sem nenhum espaço vazio entre eles.



> Fonte da imagem: openframeworks.cc

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Tamanho de um Vetor: `sizeof`

```c
#include <stdio.h>

int main() {

    int numeros[5];

    printf("%lu", sizeof(numeros));

    return 0;
}
```

Se `int` ocupa 4 bytes:

```
5 x 4 = 20 bytes
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Erros Comum: Acesso Fora do Limite

```c
int v[3];

v[3] = 10;
```

Índices válidos:

```
0, 1, 2
```

Consequências:

- Comportamento indefinido
- Corrupção de memória

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Procurando um Valor em um Vetor

```c
#include <stdio.h>

int main() {
    int numeros[5] = {10, 20, 30, 40, 50};
    int busca = 30;

    for(int i = 0; i < 5; i++) {

        if(numeros[i] == busca) {
            printf("Encontrado!");
        }
    }

    return 0;
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Encontrando o Maior Valor em um Vetor

```c
#include <stdio.h>

int main() {
    int numeros[5] = {3, 7, 2, 9, 1};
    int maior = numeros[0];

    for(int i = 1; i < 5; i++) {

        if(numeros[i] > maior) {
            maior = numeros[i];
        }
    }

    printf("Maior = %d", maior);

    return 0;
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Conclusão

Arrays são fundamentais na programação em C porque:

- Permitem armazenar múltiplos dados
- Tornam algoritmos mais eficientes
- São base para estruturas mais avançadas:
  - matrizes
  - strings
  - listas
  - estruturas dinâmicas

Próximos Tópicos: Matrizes e Strings em C

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Dúvidas e Discussão

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exercício e Questões

Questão 1

Um array **float temperaturas[7]** foi declarado. Se um programador tenta acessar o elemento na posição **temperaturas[8]**, qual é a consequência mais provável, do ponto de vista da segurança em C?

A) O programa irá compilar com erro e não rodará.
B) O compilador emitirá um aviso (warning), mas o programa funcionará corretamente.
C) O programa tentará ler/escrever fora dos limites alocados, resultando em Comportamento Indefinido (Undefined Behavior).
D) O sistema operacional irá travar imediatamente.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Questão 2

Qual é o erro no código abaixo?

```
int v[5];

v[5] = 10;
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exercício 1

Crie um programa que:

- Leia um vetor com 8 posições
- Conte quantos números são pares
- Conte quantos números são ímpares

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exercício 2

Crie um programa que:

- Leia 10 números
- Armazene em um vetor
- Ordene os números em ordem crescente
- Exiba o vetor ordenado

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br