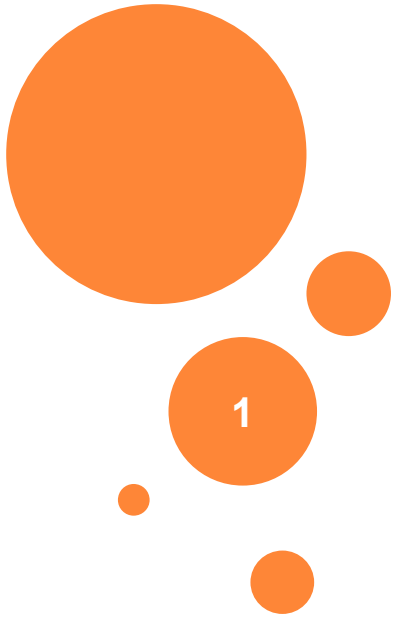


Linguagem C: Array: vetores

Prof. Humberto Razente

Sala 1B144

POR QUE USAR ARRAY?

- As variáveis declaradas até agora são capazes de armazenar apenas um valor por vez
  - Sempre que tentamos armazenar um novo valor dentro de uma variável, o valor antigo é sobrescrito e, portanto, perdido

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
   float x = 10;
   printf("x = %f\n",x);
   x = 20;
   printf("x = %f\n",x);
   system("pause");
   return 0;
}
```

Saída

x = 10.000000
x = 20.000000

ARRAY

- Array ou “vetor” é a forma mais familiar de dados estruturados

- Basicamente, um array é uma sequência de elementos do mesmo tipo, onde cada elemento é identificado por um índice

  - A ideia de um array ou “vetor” é bastante simples: **criar um conjunto de variáveis do mesmo tipo utilizando apenas um nome**

ARRAY - PROBLEMA

- Imagine o seguinte problema
  - leia as notas de uma turma de cinco estudantes e **depois** imprima as notas que são maiores do que a média da turma

- Um algoritmo para esse problema poderia ser o mostrado a seguir

ARRAY - SOLUÇÃO

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
  float n1,n2,n3,n4,n5;
  printf("Digite a nota de 5 estudantes: ");
  scanf("%f",&n1);
  scanf("%f",&n2);
  scanf("%f",&n3);
  scanf("%f",&n4);
  scanf("%f",&n5);
  float media = (n1+n2+n3+n4+n5)/5.0;
  if(n1 > media) printf("nota: %f\n",n1);
  if(n2 > media) printf("nota: %f\n",n2);
  if(n3 > media) printf("nota: %f\n",n3);
  if(n4 > media) printf("nota: %f\n",n4);
  if(n5 > media) printf("nota: %f\n",n5);

  return 0;
}
```

ARRAY

- O algoritmo anterior apresenta uma solução possível para o problema apresentado

- Porém, essa solução é inviável para grandes quantidades de alunos

  - Imagine se tivéssemos que processar as notas de 100 alunos

Array

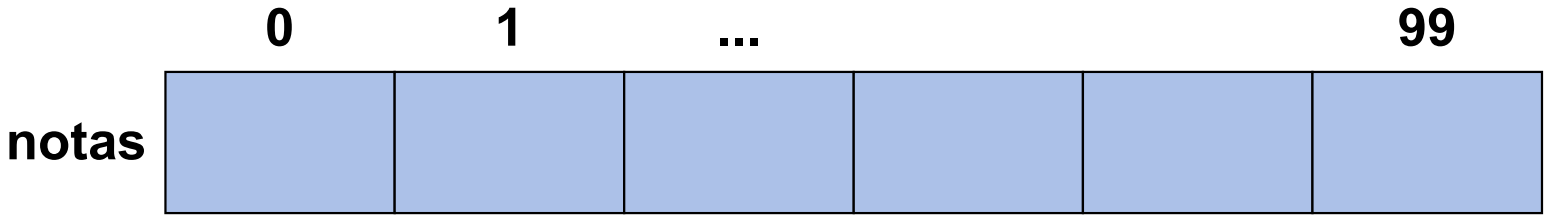
- Para 100 alunos, precisamos de:
  - Uma variável para armazenar a nota de cada aluno
    - **100 variáveis**
  - Um comando de leitura para cada nota
    - **100 scanf()**
  - Um somatório de **100 notas**
  - Um comando de teste para cada aluno
    - **100 comandos if**
  - Um comando de impressão na tela para cada aluno
    - **100 printf()**

ARRAY - DEFINIÇÃO

- As variáveis têm relação entre si
  - todas armazenam notas de alunos

- Podemos declará-las usando apenas um nome para todos os 100 alunos

  - notas: conjunto de 100 valores acessados por um índice
  - Isso é um **vetor (array)** !

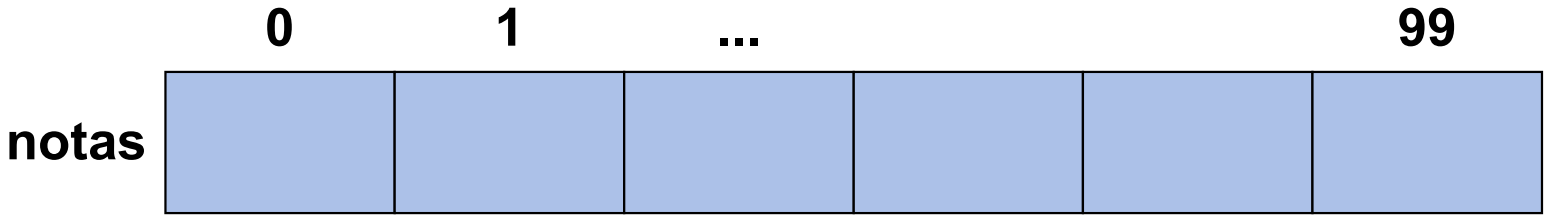


ARRAY - DECLARAÇÃO

- Arrays são agrupamentos de dados adjacentes na memória. Declaração:
  - *tipo_dado nome_array[tamanho];*

- O comando acima define um array de nome **nome_array,** capaz de armazenar **tamanho** elementos adjacentes na memória do tipo **tipo_dado**
  - Exemplo:

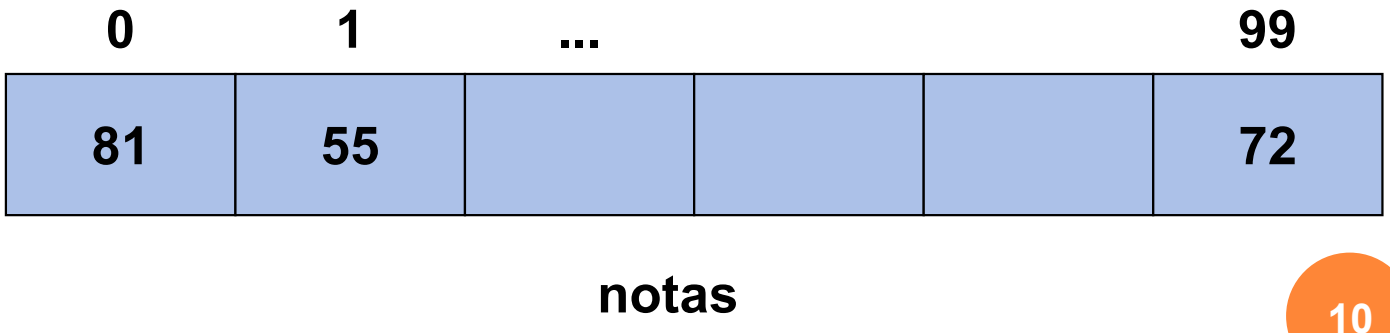
int notas[100];



ARRAY - DECLARAÇÃO

- Em um array, os elementos são acessados especificando o índice desejado entre **colchetes [ ]**
- A numeração começa sempre do zero
- Isto significa que um array de 100 elementos terá índices de 0 a 99:
  - notas[0], notas[1], notas[2], ..., notas[99]

```
int notas[100];
notas[0] = 81;
notas[1] = 55;
...
notas[99] = 72;
```



ARRAY - DEFINIÇÃO

- Observação
  - Se o usuário digitar mais de 100 elementos em um array de 100 elementos, o programa tentará ler normalmente
  - Porém, o programa os armazenará em uma parte não reservada de memória, pois o espaço reservado para o array foi para somente 100 elementos
  - Isto pode resultar nos mais variados erros durante a execução do programa

ARRAY = VARÍAVEL

- Cada elemento do array tem todas as características de uma variável e pode aparecer em expressões e atribuições (respeitando os seus tipos)
  - notas[2] = x + notas[3];
  - if (notas[2] > 60)

- Exemplo: somar todos os elementos de notas:

```
int soma = 0;
for(i=0;i < 100; i++)
    soma = soma + notas[i];
```

Percorrendo um array

- Podemos usar um comando de repetição (for, while e do-while) para percorrer um array
- Exemplo: soma dos elementos de um array de 5 elementos

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
    int lista[5] = {3,51,18,2,45};
    int i, soma = 0;
    for(i = 0; i < 5; i++)
        soma = soma + lista[i];

    printf("soma = %d\n",soma);

    return 0;
}
```

Variáveis		
i	**lista[i]**	**soma**
		0
0	3	3
1	51	54
2	18	72
3	2	74
4	45	119
5		

ARRAY - CARACTERÍSTICAS

- Características básicas de um Array

  - Estrutura **homogênea**, isto é, é formado por elementos do mesmo tipo

  - Todos os elementos da estrutura são igualmente acessíveis, isto é, o tempo e o tipo de procedimento para acessar qualquer um dos elementos do array é o mesmo

  - Cada elemento do array tem um índice próprio segundo sua posição no conjunto

ARRAY - PROBLEMA

- Voltando ao problema anterior

  - leia as notas de uma turma de cinco estudantes e depois imprima as notas que são maiores do que a média da turma.

ARRAY - SOLUÇÃO

- Um algoritmo para esse problema usando array:

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
    float notas[5];
    int i;
    printf("Digite as notas dos estudantes\n");
    for(i = 0; i < 5; i++){
        printf("Nota do estudante %d:",i);
        scanf("%f",&notas[i]);
    }
    float media = 0;
    for(i = 0; i < 5; i++)
        media = media + notas[i];
    media = media / 5;

    for(i = 0; i < 5; i++)
        if(notas[i] > media)
            printf("Notas: %f\n",notas[i]);

    return 0;
}
```

ARRAY - SOLUÇÃO

- Se ao invés de 5, fossem 100 alunos?

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
    float notas[100];
    int i;
    printf("Digite as notas dos estudantes\n");
    for(i = 0; i < 100; i++){
        printf("Nota do estudante %d:",i);
        scanf("%f",&notas[i]);
    }
    float media = 0;
    for(i = 0; i < 100; i++)
        media = media + notas[i];
    media = media / 100;

    for(i = 0; i < 100; i++)
        if(notas[i] > media)
            printf("Notas: %f\n",notas[i]);

    return 0;
}
```

Exercício

- Para um array A com 5 números inteiros, formular um algoritmo que determine o maior elemento deste array

Exercício - Solução

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
  int i, A[5] = {3,18,2,51,45};
  int ma = A[0];

  for(i=1; i<5; i++){
    if(ma < A[i])
        ma = A[i];
  }

  printf("Maior = %d\n", ma);

  return 0;
}
```

Variáveis		
i	ma	A[i]
0	3	3
1	18	18
2	18	2
3	51	51
4	51	45
5		

COPIANDO UM ARRAY

- Não se pode fazer atribuição de arrays inteiros, apenas de suas posições individualmente

```c
#include <stdio.h>
#include <stdlib.h>
int main(){
  int v[5] = {1,2,3,4,5};
  int v1[5];

  v1 = v; //ERRADO!

  int i;
  for(i=0; i<5; i++)
    v1[i] = v[i]; //CORRETO

  return 0;
}
```

TAMANHO DO ARRAY

- Em ANSI C, arrays têm tamanho definido em tempo de compilação:

  - uso de um #define

```
#define TAMANHO 50
int notas[TAMANHO];
```

  - uso de uma constante

```
const int tamanho = 50;
int notas[tamanho];
```

- Em C99, arrays também podem ter tamanho definido em tempo de execução:

```
int tamanho;
scanf("%d", &tamanho);
int notas[tamanho];
```

EXERCÍCIOS

- 1. Escreva um algoritmo que leia 10 valores inteiros e armazene-os em um vetor. Após a digitação do último valor, imprima os elementos pares.

- 2. Escreva um algoritmo que leia 15 valores reais e armazene-os em um vetor. Após a entrada do último valor, imprima-os na sequência contrária

- 3. Escreva um algoritmo que leia 20 elementos em um vetor de inteiros denominado A. Construa um outro vetor B, de mesma dimensão de A, com seus elementos sendo a multiplicação do elemento correspondente de A por 3. Mostre os elementos de B.

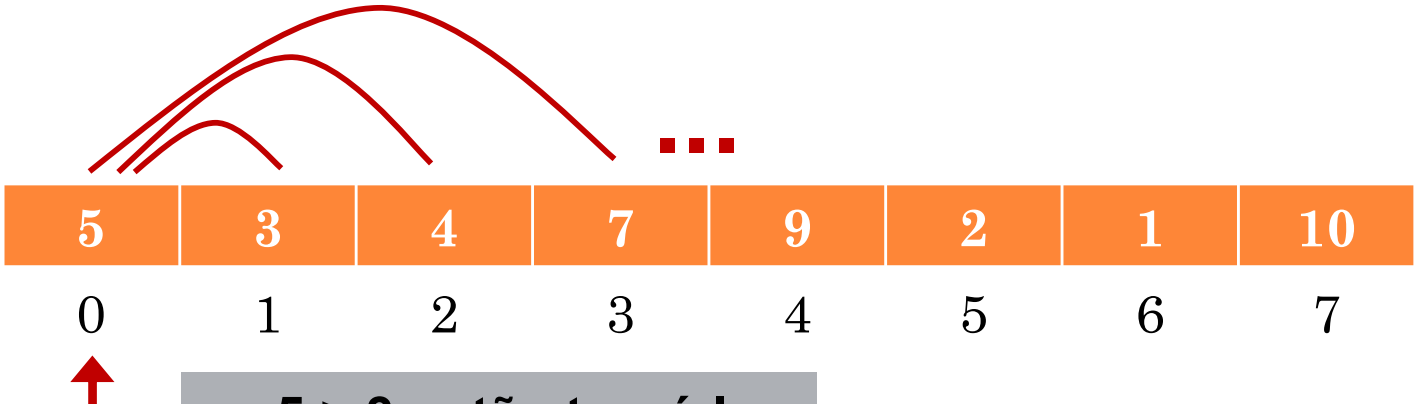
EXERCÍCIOS

- 4. Escreva um algoritmo que leia um vetor de N elementos. Após a digitação do último valor, encontre e mostre o maior elemento.

- 5. Leia 20 elementos em um vetor A e construa um novo vetor B com elementos de A, porém invertidos, ou seja, o primeiro elemento de A passa a ser o último elemento de B, e assim por diante. Depois de criado o vetor B com os valores, imprima os 2 vetores.

Problema: Ordenação de Dados

- A ordenação dos valores em um vetor pode ser realizada comparando-se cada elemento com todos os outros que ainda não foram ordenados, efetuando a troca quando o elemento sob análise for maior que o outro elemento. Escrever um algoritmo para ordenar um vetor. A cada elemento já ordenado, imprimir o elemento. Executar em modo de depuração para avaliar o algoritmo.



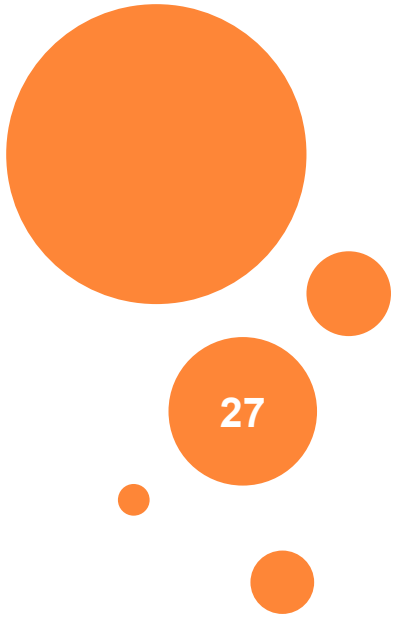
Exercício 6

- Escreva um algoritmo que ordene um vetor de números inteiros

Material Complementar

- Vídeo Aulas
  - Aula 25: Array / Vetor
  - Aula 28: Inicialização de Arrays
  - Aula 29: Somando um Array
  - Aula 30: Maior valor de um Array

- https://programacaodescomplicada.wordpress.com/indice/linguagem-c/



Linguagem C: Array: vetores e matrizes

Contém slides originais gentilmente disponibilizados pelo Prof. André R. Backes (UFU)