

# Minicurso de C



### Minicurso de C

Módulo 5 – Vetores e Matrizes

Fernando Morgado Pires Neto Renan Ricoldi Fróis Pedro



## Objetivo



Introduzir o conceito de Estrutura de Dados Ensinar vetores e matrizes na linguagem C

### **Estrutura de Dados**



• Área que estuda as diferentes maneiras de organizar um conjunto de valores (dados).

### **Estrutura de Dados**



• Existem diversos tipos de estruturas de dados, entre elas estão os vetores e as matrizes.



		•		A B 4
mΩ	mo	ric	1 K	AM
шс		LL	4 IN	Alti

813712	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3



```
#include <stdio.h>
int main(){
   int resultado, valor1, valor2;
   resultado = 2 + 2;
   return 0;
}
```

813712	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3



```
#include <stdio.h>
int main(){
   int resultado, valor1, valor2;
   resultado = 2 + 2;
   return 0;
}
```

813712	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3







```
#include <stdio.h>
int main(){
   int resultado, valor1, valor2;
   resultado = 2 + 2;
   return 0;
}
```

4	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3



valor	4	31232	32	-120	4213	32134	323	12381	3123	4848
endereço de memória	А	В	С	D	Е	F	0	1	2	3
variável	resultado	valor1	valor2							



- Criar variáveis é ok para quando queremos armazenar dados pontuais.
- Mas e para quando precisarmos armazenar grandes quantidades de dados?

### **Problema**



• Exemplo:

Faça um programa que armazene as notas de 8 alunos e depois as imprima na tela.

### **Problema**



```
vintenotas.c
#include <stdio.h>
int main(){
    int nota1, nota2, nota3, nota4, ..., nota8;
    scanf("%d", &nota1);
    scanf("%d", &nota2);
    scanf("%d", &nota3);
return 0;
```



### Vetor



 Possibilita que você possa armazenar uma grande quantidade de valores do mesmo tipo em apenas uma variável.



• Como declarar um vetor:

tipo nome\_da\_variavel[quantidade]



```
vintenotas.c
#include <stdio.h>
int main(){
    int notas[8];
return 0;
```



```
#include <stdio.h>

int main(){
   int notas[8];
   ...
}
```

2948	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3



```
#include <stdio.h>

int main(){
   int notas[8];
   ...
}
```

2948	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	E	F	0	1	2	3







Como inicializar um vetor com valores:

```
tipo nome_da_variavel[] = { valor1, valor2, ... valorN }
tipo nome_da_variavel[5] = { valor1, valor2 }
```



```
• • incializacao.c
#include <stdio.h>
int main(){
    int notas[] = \{6, 8, 9, 10\};
    int medias [4] = \{10, 8\}
     . . .
```



```
inicializacao.c

#include <stdio.h>

int main(){

   int notas[] = {6, 8, 9, 10};
   int medias[4] = {10, 8}

...
}
```

2948	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3



```
int main(){
   int notas[] = {6, 8, 9, 10};
   int medias[4] = {10, 8}
```

2948	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3







```
int main(){
   int notas[] = {6, 8, 9, 10};
   int medias[4] = {10, 8}
```

6	8	9	10	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3







```
int main(){
   int notas[] = {6, 8, 9, 10};
   int medias[4] = {10, 8}
```

6	8	9	10	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3







```
int main(){
   int notas[] = {6, 8, 9, 10};
   int medias[4] = {10, 8}
```

6	8	9	10	10	8	0	0	3123	4848
А	В	С	D	Е	F	0	1	2	3







Como atribuir um valor a uma posição:

```
nome_da_variavel[indice] = valor
       scanf("%tipo", &nome_da_variável[indice])
tipo nome_da_variavel[5];
nome_da_variavel[0] = 10; -> a posição (índice) sempre começa em zero
nome_da_variavel[1] = 20;
nome_da_variavel[2] = 10;
nome_da_variavel[3] = 10;
nome_da_variavel[4] = 10;
```



```
inicializacao.c

#include <stdio.h>

int main(){

   int notas[] = {6, 8, 9, 10};
   int medias[4] = {10, 8}

...
}
```

6	8	9	10	10	8	0	0	3123	4848
А	В	С	D	E	F	0	1	2	3



```
inicializacao.c

#include <stdio.h>

int main(){

   notas[0] = 7;
   media[3] = 5;

}
```

6	8	9	10	10	8	0	0	3123	4848
А	В	С	D	Е	F	0	1	2	3



```
#include <stdio.h>

int main(){

   notas[0] = 7;
   media[3] = 5;

}
```

7	8	9	10	10	8	0	0	3123	4848
А	В	С	D	Е	F	0	1	2	3





```
#include <stdio.h>

int main(){

   notas[0] = 7;
   media[3] = 5;

...
}
```

7	8	9	10	10	8	0	0	3123	4848
А	В	С	D	E	F	0	1	2	3



```
#include <stdio.h>

int main(){

   notas[0] = 7;
   media[3] = 5;
}
```

7	8	9	10	10	8	0	5	3123	4848
А	В	С	D	E	F	0	1	2	3



valor	7	8	9	10	10	8	5	0	3123	4848
endereço de memória	А	В	С	D	E	F	0	1	2	3
como acessar o valor	notas[0]	notas[1]	notas[2]	notas[3]	media[ <b>0</b> ]	media[1]	media[2]	media[3]	XXXXX	XXXXX



• Exemplo:

Faça um programa que armazene as notas de 8 alunos e depois as imprima na tela.

### **Exercícios - Vetor**



1. Faça um programa que **armazene 10 valores** inseridos pelo usuário, e depois os **imprima na tela em ordem reversa**.



2. Na estatística, os cálculos de média, moda e mediana são comuns. Faça um programa que **armazene 10 valores**, e imprima os valores de **média, moda e mediana** entre eles.



3. Renan é um professor que da aula para 20 alunos em uma turma de programação. Você, como é um aluno aplicado, decidiu ajudar Renan a calcular a média de notas da turma durante um bimestre.

Faça um programa que armazene as 20 notas, faça a média entre elas, e imprima o valor na tela. Também mostre qual foi a maior e menor nota da turma.

<sup>\*</sup>Fazer esse exercício utilizando vetor!



4. A teoria dos conjuntos é um ramo extremamente importante na matemática. Faça um programa que receba 8 valores e os armazene em um vetor A. Depois, receba outros 10 valores e os armazene em um vetor B.

Imprima na tela os valores da <u>união</u> e <u>intersecção</u> dos vetores A e B.



5. Faça um programa que receba 10 valores e os armazene em um vetor. Depois, ordene o vetor em ordem crescente e o imprima na tela.



• Uma **string** é uma cadeia de caracteres (**char**) armazenadas em um vetor.



```
#include <stdio.h>

int main(){
    char palavra[] = "banana";
}
```

2948	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3



```
#include <stdio.h>

int main(){
    char palavra[] = "banana";
}
```

2948	31232	32	-120	4213	32134	323	12381	3123	4848
А	В	С	D	E	F	0	1	2	3







```
#include <stdio.h>

int main(){
    char palavra[] = "banana";
}
```

b	а	n	а	n	а	323	12381	3123	4848
А	В	С	D	Е	F	0	1	2	3







```
#include <stdio.h>

int main(){
    char palavra[] = "banana";
}
```

b	а	n	а	n	а	\0	12381	3123	4848
А	В	С	D	E	F	0	1	2	3







caractere	b	а	n	а	n	а	\0
valor na tabela ASCII	98	97	110	97	110	97	Ø
endereço de memória	А	В	С	D	E	F	0
como acessar o valor	palavra[ <b>0</b> ]	palavra[1]	palavra[2]	palavra[3]	palavra[ <b>4</b> ]	palavra[ <b>5</b> ]	palavra[ <b>6</b> ]



- Manipular nativamente strings em C pode ser muito complicado.
- Para isso, a linguagem possui a biblioteca string.h para manipulações mais complexas entre strings.

# **Exercícios - String**



6. <u>Sem utilizar a biblioteca string.h,</u> faça um programa que compare duas senhas digitadas pelo usuário. Caso elas forem iguais, imprima na tela "Senhas iguais". Caso contrario, imprima "Senhas diferentes".



• O conceito de **matriz** é similar ao já conhecido na matemática



1 2 3



1	2	3
4	5	6
7	8	9



• Como declarar uma matriz:

tipo nome\_da\_variavel[quantidadeLinhas][quantidadeColunas]



int matriz[3][2]

0	0
0	0
0	0



matriz[1][0] = 4

0	0
0	0
0	0



$$matriz[1][0] = 4$$

0	0
0	0
0	0

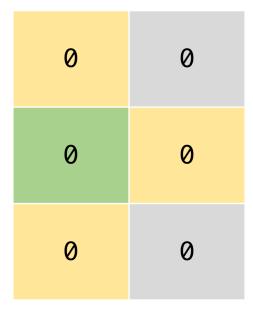


matriz[1][0] = 4

0	0
0	0
0	0

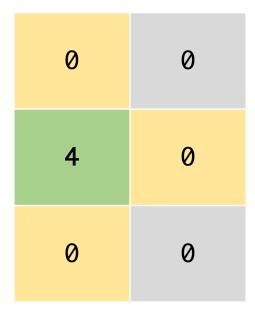


matriz[1][0] = 4





$$matriz[1][0] = 4$$



## **Exercícios - Matriz**



7. Preencha uma matriz 3x3 com valores inseridos pelo usuário, e depois imprima os valores na tela.

## **Exercícios - Matriz**



8. Preencha uma matriz 3x3 com valores inseridos pelo usuário. Verifique se a matriz inserida é uma **matriz** diagonal.

6	0	0
0	13	0
0	0	9

# Fim!

# **Dúvidas?**

# Obrigado pela atenção!

