FCT/Unesp – Presidente Prudente Departamento de Matemática e Computação

ATP I

Vetores

Prof. Dr. Danilo Medeiros Eler danilo.eler@unesp.br

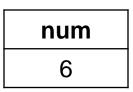




Variáveis

- Manipulamos valores armazenados em memória por meio de variáveis
- Até o momento, vimos como acessar um único valor por meio de uma variável
- Por exemplo:

```
int num = 6;
printf("%d\n",num);
```







- Para armazenar mais de um valor é necessário mais de uma variável
- Entretanto, isso é impraticável quando a quantidade de valores aumenta
- Por isso, existem os vetores para representarem um conjunto de valores armazenados na memória
 - Podem ser acessados por meio de uma variável
 - Vetor também é conhecido como arranjo
 - Em inglês array





- Na linguagem C podemos especificar um vetor desse modo: int valores[50];
- No exemplo, criamos uma variável denominada valores
- Essa variável acessa uma área de memória reservada para armazenar 50 valores do tipo inteiro
- Cada valor está em uma posição específica do vetor
 - A posição também é chamada de índice

valores	
Índice	Dado
0	
1	
2	
3	
4	
5	
6	
47	
48	
49	





- As posições de um vetor variam de 0 a N-1
- As posições são acessíveis dessa maneira:

```
valores[4] = 5;
valores[48] = 7;
```

 Para exibir todos os valores é preciso utilizar uma repetição

```
for(int i=0; i<50; i++){
  printf("%d ", valores[i]);
}</pre>
```

valores	
Índice	Dado
0	
1	
2	
3	
4	5
5	
6	
47	
48	7
49	





```
int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){
    scanf("%d",&num);
    valores[i] = num;
}</pre>
```





```
int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){
    scanf("%d",&num);
    valores[i] = num;
}</pre>
```

num

valores		
Índice	Dado	
0		
1		
2		
3		
4		





int valores[5], num;

num

printf("Digite 5 valores: ");
for(int i=0; i<5; i++){</pre>

scanf("%d",&num);

valores[i] = num;

}

Digite 5 valores:

valores		
Índice	Dado	
0		
1		
2		
3		
4		



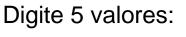


```
int valores[5], num;
printf("Digite 5 valores: ");

→ for(int i=0; i<5; i++){
    scanf("%d",&num);
    valores[i] = num;
}</pre>
```

num	
Ì	
0	

valores	
Índice	Dado
0	
1	
2	
3	
4	





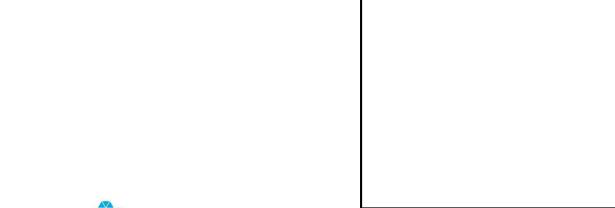


```
int valores[5], num;
printf("Digite 5 valores: ");

→ for(int i=0; i<5; i++){
    scanf("%d",&num);
    valores[i] = num;
}</pre>
Digite 5 valores:
```

num		
)	

valores	
Índice	Dado
0	
1	
2	
3	
4	





int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){</pre>

	num	
	5	
_		

valores		
Índice	Dado	
0		
1		
2		
3		
4		

→ scanf("%d",&num); valores[i] = num;

Digite 5 valores:	
5	





```
int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){
    scanf("%d",&num);
    Digite</pre>
```

num	
5	
i	
0	

valores		
Índice	Dado	
0	5	
1		
2		
3		
4		

	valores[i] = num;
}	

Digite 5 valores:	
5	





```
int valores[5], num;
printf("Digite 5 valores: ");
for/int i_O. i = E. i. . \(
```

```
num
```

valores		
Índice	Dado	
0	5	
1		
2		
3		
4		

—	$O((1111) = 0, 1 < 5, 1 + +){$
	scanf("%d",#);
	valores[i] = num;
١	

Digite 5 valores:	
5	





int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){</pre>

num	
8	
i	

valores		
Índice	Dado	
0	5	
1		
2		
3		
4		

scanf("%d",&num);
valores[i] = num;





```
int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){
    scanf("%d",&num);
    Digite
    valores[i] = pum;</pre>
```

num	
8	
i	
1	

valores		
Índice	Dado	
0	5	
1	8	
2		
3		
4		

	valores[i] = num;
}	

Digite 5 valores: 5	
8	





```
int valores[5], num;
printf("Digite 5 valores: ");
```

→ for(int i=0; i<5; i++){

scanf("%d",&num);

valores[i] = num;
}

num		
8		
	i	

Digite 5 valores:	
5	
8	

valores		
Índice	Dado	
0	5	
1	8	
2		
3		
4		





```
int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){</pre>
```

num	
0	
i	
2	

valores		
Índice	Dado	
0	5	
1	8	
2		
3		
4		

→	scanf("%d",#);
	valores[i] = num;
١	

Digite 5 valores: 5	
8	
0	





num	
0	
i	
2	

valores			
Índice	Dado		
0	5		
1	8		
2	0		
3			
4			

	,	וט
Դ;		5
•		8
		0





```
int valores[5], num;
printf("Digite 5 valores: ");
```

$Or(Int I=0; I<5; I++){$
scanf("%d",#);
valores[i] = num;

num	
0	
i	
3	

Digite 5 valores:
5
8
0

valores			
Índice	Dado		
0	5		
1	8		
2	0		
3			
4			





Vetore<u>s</u>

int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){</pre>

num	
52	
i	

3

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3		
4		
·		

—	scanf("%d",#);
	valores[i] = num;
-	

Digite 5 valores:
5
8
0
52





int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){
 scanf("%d",&num);
 Digite</pre>

num	
52	
i	
3	

valores		
Dado		
5		
8		
0		
52		

valores[i]	=	num;
•		

Digite 5 valores:	
5	
8	
0	
52	





```
int valores[5], num;
printf("Digite 5 valores: ");
```

for(int i=0; i<5; i++){
 scanf("%d",&num);
 valores[i] = num;</pre>

num
52
i

Digite 5 valores:
5
8
0
52

valores	
Índice	Dado
0	5
1	8
2	0
3	52
4	





int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){</pre>

num	
44	
	_

i	
4	

•	or(int i=0, 130, 11.7)
—	scanf("%d",#);
	valores[i] = num;
١	

Digite 5 valores:
5
8
0
52
44

valores	
Índice	Dado
0	5
1	8
2	0
3	52
4	





int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){
 scanf("%d",&num);
 Digite</pre>

num	
44	
i	
4	

valores	
Índice	Dado
0	5
1	8
2	0
3	52
4	44
·	·

—	valores[i] = num;
}	

Digite 5 valores:
5
8
0
52
44





```
int valores[5], num;
```

```
44
    printf("Digite 5 valores: ");
\rightarrow for(int i=0; i<5; i++){
```

	scanf("%d",#);
	valores[i] = num;
}	

-
Digite 5 valores:
5
8
0
52
44

num

valores		
Dado		
5		
8		
0		
52		
44		



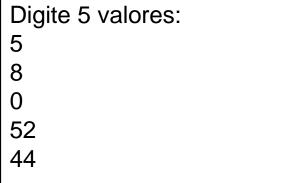


```
int valores[5], num;
printf("Digite 5 valores: ");
for(int i=0; i<5; i++){
 scanf("%d",&num);
 valores[i] = num;
```

num	
44	
i	
5	

44	
i	
5	

valores		
Dado		
5		
8		
0		
52		
44		







int valores[5], num;

num 44

printf("Valores armazenados: ");

i 5

for(int i=0; i<5; i++){
 printf("%d ", valores[i]);</pre>

}

Va	alores armazenados:	

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	





int valores[5], num;
printf("Valores armazenados: ");

→ for(int i=0; i<5; i++){
 printf("%d ", valores[i]);</pre>

	num	
	44	
i		
	0	

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

Valores armazenados:

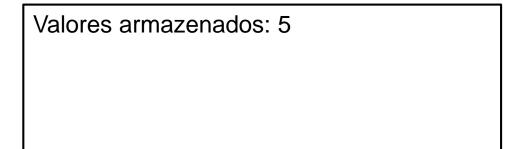




int valores[5], num;
printf("Valores armazenados: ");
for(int i=0; i<5; i++){
 printf("%d ", valores[i]);</pre>

	num	
	44	
i		
	0	

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	



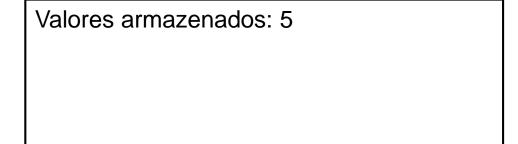




int valores[5], num; printf("Valores armazenados: "); \rightarrow for(int i=0; i<5; i++){ printf("%d ", valores[i]);

	num
	44
,	_
,	i
	1

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	



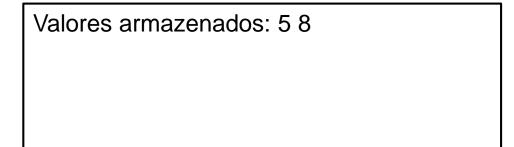




int valores[5], num;
printf("Valores armazenados: ");
for(int i=0; i<5; i++){
 printf("%d ", valores[i]);</pre>

num	
44	
i	$\overline{\ \ }$
 1	1

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	







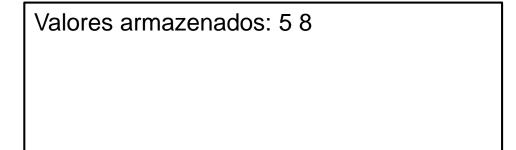
int valores[5], num;

printf("Valores armazenados: ");

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

printf("%d ", valores[i]);

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	







int valores[5], num;
printf("Valores armazenados: ");
for(int i=0; i<5; i++){
 printf("%d ", valores[i]);</pre>

num	
44	
i	
2	

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

Valores armazenados: 5 8 0





int valores[5], num;

printf("Valores armazenados: ");

i

→ for(int i=0; i<5; i++){

printf("%d ", valores[i]);

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

Valores armazenados: 5 8 0





int valores[5], num;
printf("Valores armazenados: ");
for(int i=0; i<5; i++){
 printf("%d ", valores[i]);</pre>

num	
44	
i	
3	

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

Valores armazenados: 5 8 0 52





int valores[5], num;
printf("Valores armazenados: ");

→ for(int i=0; i<5; i++){</pre>

num	
44	
i]
	4

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

printf("%d ", valores[i]);

Valores armazenados: 5 8 0 52





int valores[5], num;
printf("Valores armazenados: ");
for(int i=0; i<5; i++){
 printf("%d ", valores[i]);</pre>

	num	
	44	
,	i	
	4	

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

Valores armazenados: 5 8 0 52 44





int valores[5], num;

printf("Valores armazenados: ");

→ for(int i=0; i<5; i++){

printf("%d ", valores[i]);

num
44
i
5

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

Valores armazenados: 5 8 0 52 44





int valores[5], num;

printf("Valores armazenados: ");

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

printf("%d ", valores[i]);

num		
	44	
•		
,	i	
	5	

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	

Valores armazenados: 5 8 0 52 44





int valores[5], num;

→ printf("Vetor posição 1: %d",valores[1]); vetor[1] = 9;

printf("Vetor posição 1: %d",valores[1]);

Vetor posição 1: 8

valores		
Índice	Dado	
0	5	
1	8	
2	0	
3	52	
4	44	





int valores[5], num; printf("Vetor posição 1: %d",valores[1]);

→ vetor[1] = 9; printf("Vetor posição 1: %d",valores[1]);

Vetor posição 1: 8	

valores		
Índice	Dado	
0	5	
1	9	
2	0	
3	52	
4	44	





int valores[5], num; printf("Vetor posição 1: %d",valores[1]); valores[1] = 9;

printf("Vetor posição 1: %d",valores[1]);

Vetor posição 1: 8 Vetor posição 1: 9

valores		
Índice	Dado	
0	5	
1	9	
2	0	
3	52	
4	44	





Exemplos

- 1) Ler e armazenar elementos em um vetor. Em seguida, exibir o vetor e a soma.
- 2) Preencher um vetor com a soma acumulada de 1 a N.
- 3) Preencher um vetor com N elementos da sequência de Fibonacci. Considere N>2.
- 4) Preencher um vetor com N valores aleatórios de 0 a 9.





Exercícios

Fazer a lista de exercícios.





Bibliografia Básica

- 1. ASCENCIO, A. F. G.; CAMPOS, E. A. V. C. **Fundamentos da programação de computadores**: algoritmos, pascal e C/C++. Pearson Prentice Hall, 2003. 355p.
- 2. KERNINGHAN, B. W.; Ritchie, D. M. **C**: a Linguagem de Programação padrão ANSI. Rio de Janeiro:Editora Campus, 1990. 289p.
- 3. KERNINGHAN, B. W.; Pike, R. **A Prática de Programação**. Rio de Janeiro: Editora Campus, 2000, 280p.
- 4. LOPES, A.; GARCIA, G. **Introdução à Programação**: 500 exercícios resolvidos. Rio de Janeiro: Editora campus, 2002. 469p.
- 5. MANZANO, J. A. N. G.; OLIVEIRA, J. F. **Algoritmos**: lógica para desenvolvimento de programação de computadores. 10^a ed. Editora Érica Ltda. 2000, 236p.
- 6. MIZRAHI, V. V. **Treinamento em Linguagem C Curso Completo Módulo 1**. São Paulo: MAKRON Books do Brasil Editora Ltda, 1990. 241p.
- 7. MEDINA, M.; FERTIG, C. **Algoritmos e programação**: teoria e prática. São Paulo: Novatec Editora. 384p. 2005.
- 7. PUGA, S.; RISSETTI, G. **Lógica de programação e estruturas de dados**: com aplicações em JAVA. São Paulo: Pearson Education do Brasil, 254p. 2004.
- 8. SCHILDT H. **C Completo e Total**. 3ª ed. São Paulo: MAKRON Books do Brasil editora Ltda. 1997. 827p.
- 9. XAVIER, G. F. C. Lógica de Programação. São Paulo: Editora SENAC. 1998. 378p.





Bibliografia Complementar

- 1. BROOKSHEAR, J. G. **Ciência da computação**: uma visão abrangente. 5^a ed., Bookman Editora, 2000. 499p.
- 2. CORMEN, T.H., Leiserson, C.E., Rivest R.L., Stein, C. **Algoritmos:** teoria e Prática. Rio de janeiro: Editora Campus, 2002. 916p.
- 3. PLAUGER, P. L. A Biblioteca Standard C. Rio de Janeiro: Editora Campus, 1994. 614p.
- 4. PRATA, S. **C** primer plus, 4^a ed. SAMS Publishing, 2002. 931p.
- 5. OLIVEIRA, U. **Programando em C**, vol. I fundamentos. Editora Ciência Moderna, 2008, 743p.



