



# Vetores

---

Prof. Tiago Gonçalves Botelho

# Vetores - Definição

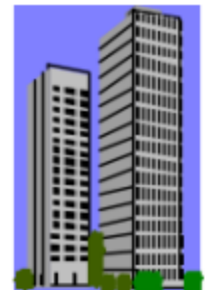
---

- Estrutura de Dados Homogênea e Estática:

- Unidimensional

- Exemplo:

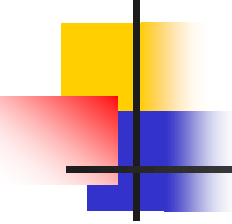
- Prédio com um apartamento por andar;



- Todos os elementos pertencentes ao mesmo tipo de dado;

- 

[illegible]



# Vetores - Definição

---

- Possuem o mesmo identificador (nome), e o que as diferencia é a sua localização dentro da estrutura (*Índices*):
  - Índices iniciam de "0" até "N-1";
  - Índices são utilizados para Recuperar/Inserir valores.

Índice	0	1	2	3	4
Vet	6	8	5	2	7



# Vetores - Declaração

---

DECLARE **nome**[**tamanho**] **TIPO**

Onde:

- **nome**: é o nome da variável do tipo vetor;
- **tamanho**: é a quantidade de variáveis que vão compor o vetor;
- **TIPO**: é o tipo básico dos dados que serão armazenados no vetor.



# Vetores - Declaração

---

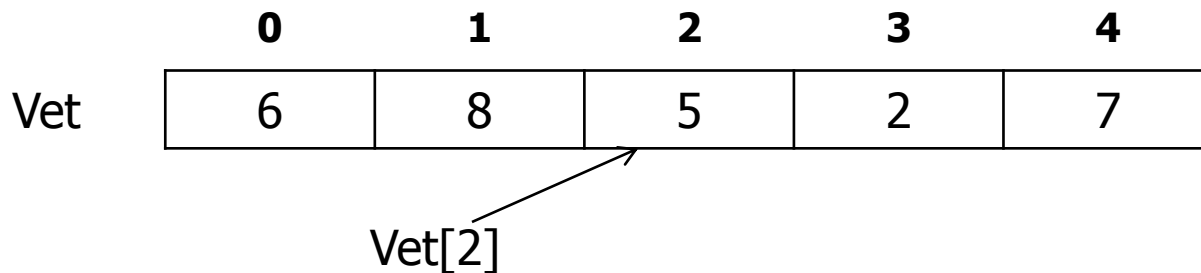
- Exemplo:

Pseudocódigo:

DECLARE X[5] INTEIRO

Em C++:

int X[5]





# Vetores – Atribuição de valores

---

- Para atribuir valores a um vetor deve-se determinar em qual(is) posição(ões) o(s) valor(es) ficará(ão) armazenado(s).

- Exemplos:

- $X[1] \leftarrow 45$

- No exemplo, o número 45 será armazenado na posição de índice 1 do vetor.*

- $X[4] \leftarrow 0$

- No exemplo, o número 0 será armazenado na posição de índice 4 do vetor.*



# Vetores – Preenchendo um vetor

---

ALGORITMO exemplo\_vetor

DECLARE X[5] INTEIRO

    i INTEIRO

INICIO

    Para i <- 0 ATÉ 4 FAÇA

        INICIO

            escreva("Digite o ",i," elemento:");

            leia(X[i]);

        FIM

...





# Vetores – Mostrando os elementos de um vetor

---

...

Para i <- 0 ATÉ 4 FAÇA

    escreva("Este é o ",i," elemento do vetor:",X[i]);

FIM.



# Vetor

---

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }
8      for(i=0; i<5; i++)
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10
11     return 0;
12 }
```

# Vetor

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }
8      for(i=0; i<5; i++)
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10
11     return 0;
12 }
```

Diagrama de anotações:

- Uma seta aponta de "i = 0" para "i=0" na linha 4.
- Uma seta aponta de "0" para "i" na linha 5.
- O texto "X[0]" está posicionado abaixo da linha 6.

# Vetor

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }
8      for(i=0; i<5; i++)
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10
11     return 0;
12 }
```

Diagrama de anotações:

- Uma seta aponta de `i = 1` para o primeiro índice `i` no loop `for(i=0; i<5; i++)`.
- Um número `1` com uma seta aponta para o primeiro índice `i` no loop `for(i=0; i<5; i++)`.
- O texto `X[1]` está escrito abaixo do primeiro loop `for`.

# Vetor

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }
8      for(i=0; i<5; i++)
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10
11     return 0;
12 }
```

Diagrama de anotações:

- Uma seta aponta de `i = 2` para o índice `i` no loop `for(i=0; i<5; i++)` na linha 4.
- Outra seta aponta de `2` para o índice `i` no `printf` na linha 5.
- O texto `X[2]` está escrito abaixo do código, alinhado com o índice `i` na linha 5.

# Vetor

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }
8      for(i=0; i<5; i++)
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10
11     return 0;
12 }
```

Diagrama de anotações:

- Uma seta aponta de `i = 3` para a condição `i < 5` na linha 4.
- Um número `3` com uma seta apontando para o índice `i` no argumento da função `scanf` na linha 6.
- O texto `X[3]` está escrito abaixo da linha 6.

# Vetor

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;          i = 4
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }                    X[4]
8      for(i=0; i<5; i++)
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10
11     return 0;
12 }
```

Diagrama de execução: Uma linha vertical com marcadores indica o fluxo de execução. Um retângulo branco está na linha 2. Um retângulo cinza está na linha 4. Uma seta aponta de "i = 4" para a condição "i < 5" na linha 4. Outra seta aponta de "4" para o índice "i" no printf da linha 5.

# Vetor

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }
8      for(i=0; i<5; i++)
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10
11     return 0;
12 }
```

Diagrama de anotação no código:

- Uma seta aponta de `i = 5` para a condição `i < 5` na linha 4.
- Outra seta aponta de `5 < 5 -> F` para a mesma condição.



# Vetor

- Preenchendo e mostrando um vetor: Código em C++

```
1  #include<bits/stdc++.h>
2  int main() {
3      int X[5], i;
4      for(i=0; i<5; i++) {
5          printf("Digite o %i elemento:\n", i);
6          scanf("%i", &X[i]);
7      }
8      for(i=0; i<5; i++) {
9          printf("Este e o %i elemento: %i\n", i, X[i]);
10     }
11     return 0;
12 }
```

Executa novamente o laço para mostrar o vetor.



# Bibliografia

---

- ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. Fundamentos da programação de computadores: algoritmos, Pascal e C/C++. São Paulo: Prentice Hall, 2002 355 p.
- MEDINA, Marco; FERTIG, Cristina. Algoritmos e programação: teoria e prática. São Paulo: Novatec, 2005. 384 p. ISBN 85-7522-073-X.